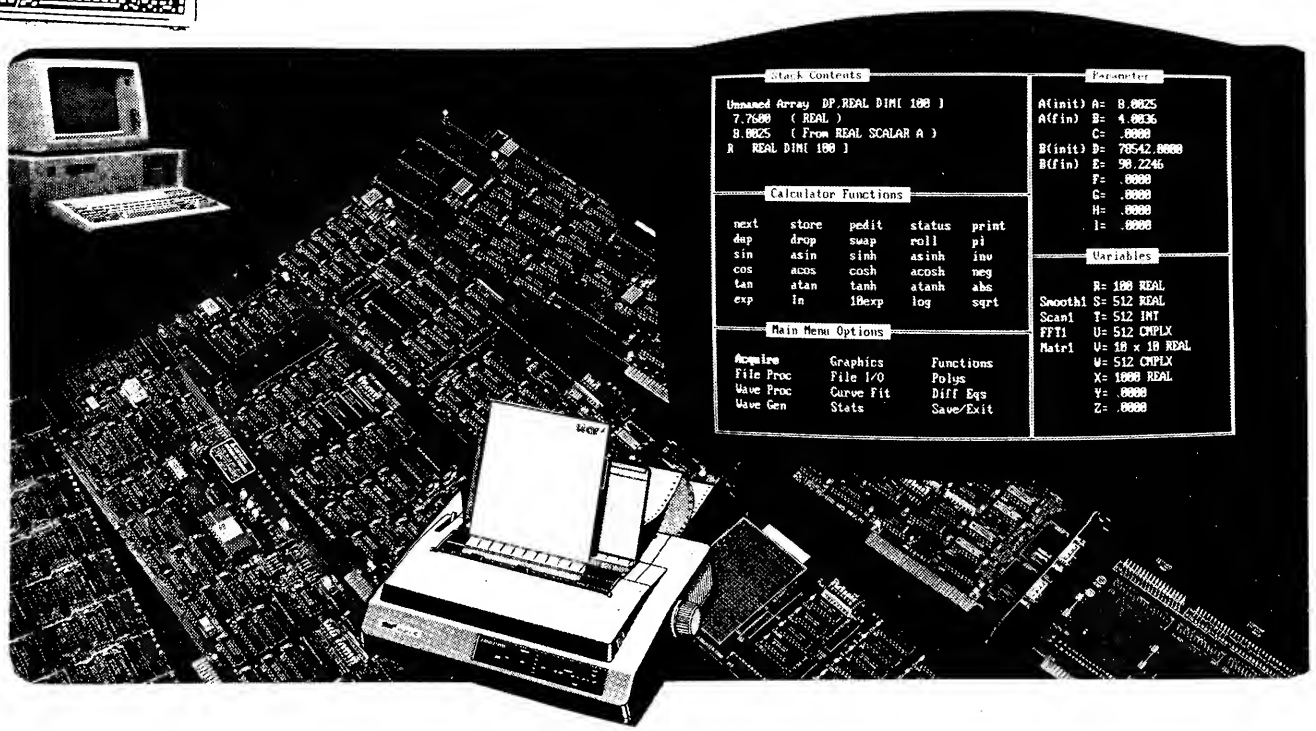
MIKROPROCESOROVÁ A VÝPOČETNÍ TECHNIKA * HARDWARE & SOFTWARE

mikroelektronika



ZELENÁ ROČENKA

je stejně jako zelené stránky časopisu Amatérské radio věnována opět tomu z elektronické zájmové činnosti, co souvisí nějak s počítači. Jsou zde převážně příspěvky z minulého ročníku soutěže Mikroprog - Mikrokonkurs, a to zejména ty rozsáhlejší, jejichž publikování v AR na pokračování by bylo nepraktické. Doplňují je drobnější konstrukční i programové výtvory, tentokrát převážně na téma "propojení počítače s tiskárnou".

Nebude asi mnoho těch, kteří do detailu "okopírují" obzvláště ty rozsáhlé příspěvky. Domníváme se ale, že není jediným účelem zveřejňovaných návodů, aby je někdo přesně okopíroval. Mnohem větší hodnotu mají jako zdroj inspirace, námětů, dílčích řešení pro úplně jiné úkoly a problémy, nebo třeba jako "učebnice" programování, ap.

Hodně radosti a poučení nad letošní "zelenou ročenkou" vám přeje vaše

redakce AR

OBSAH:

Vydalo Vydavatelství Naše vojsko. Adresa redakce: Jungmannova 24, 113 66 Praha 1, tel. 260651-7. Šéfredaktor ing. Jan Klabal, odpovědný redaktor ing. Alek Myslík. Sekretářka redakce T. Trnková, linka 355. Vytisklo Naše vojsko, s.p., závod 08, Vlastina 889/23, 162 00 Praha 6. Za původnost a správnost příspěvku ručí autor.

KLÁVESNICE A DISPLEJ

PRO JEDNOČIPOVÉ MIKROPOČÍTAČE

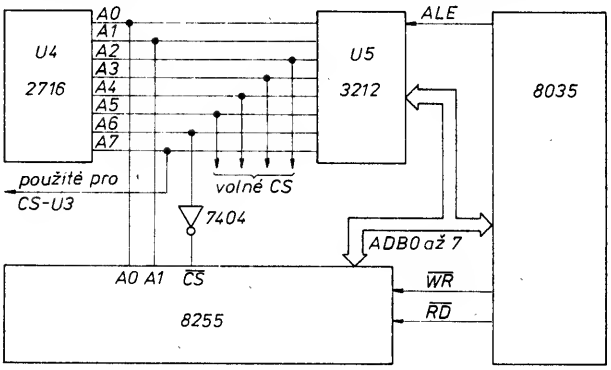
Jiří Tobola, *Oldřicha Stibora 63, 739 31 Řepiště*

V celé řadě přístrojů a jednoduchých zařízení stojíme před řešením funkční a datové klávesnice a zobrazování potřebných hodnot na sedmisegmentovém displeji.

Předložené řešení počítá s minimální technickou podporou. Vzhledem k tomu, že jsou dostupné překladače ASM48 pro celou řadu počítačů, generovat aplikační SW při podpoře základních rutin by nemělo dělat problém celé řadě zájemců o použití v nejrůznějších aplikacích řízení. Popis bude věnován hlavně programovému vybavení, které tvoří hlavní podporu aplikací. Proto výpisu textu programu je třeba věnovat zvýšenou pozornost.

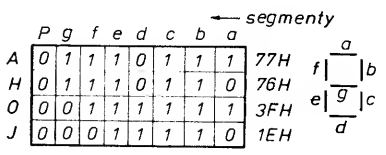
Popis zapojení pro 8035 (obr. 1)

Zobrazovací jednotka pracuje v multiplexním režimu spolu se snímáním klávesnice. Počet prvků displeje a klávesnice může být volen podle aplikace změnou inicializačních proměnných. Segmenty jednotlivých zobrazovačů jsou spojeny paralelně a připojeny na budič segmentů U2 přes omezovací rezistory 100 Ω (75 Ω). Může být použit libovolný budič pro výkonové posílení portu P1 (8286, 8287, 3212, 3216 apod.). Budiče číslic jsou tvořeny tranzistory T1 až T8 a obvodem 3212.



Obr. 2. Připojení 8255 pro rozšíření I/O prostoru (922–2)

Klávesnice v aktivním stavu spíná příslušný nulový buzený bit obvodu U3 s bity portu P2 (P24 až P27) tak, že uvádí do log.0 příslušný bit portu P2, předtím nastavený na log.1 instrukcí OUTL P2,A. Latch U3 (3212) zároveň budí příslušnou číslici displeje přes spínací tranzistory T1 až T8. Rezistory v bázích tranzistorů T1 až T8 jsou v rozsahu 1 až 2,2 kΩ. Zápis do budiče čísel (U3) používá paměťovou instrukci MOVX @RO,A. Je možné samozřejmě použít dekodér 3205, připojený za latch 3212 (U5). Do takto orientovaného I/O prostoru lze připojit obvod 8255 pro jeho rozšíření. Doporučuji také porty P1 a BUS připojit přes rezistory 10 kΩ na 5 V, pokud to bude potřebné k definici výchozího stavu. Vnější paměť programu může být tvořena dostupnými paměťmi PROM-EPROM podle rozsahu aplikace. Adresní latch 3212 (U5) tvoří adresu pro paměť programu a spolu se signálem PSEN dává kód instrukce na sběrnici.



Obr. 4. Kódování pozdravu „AHOJ" v testovacím programu (922–4)

Příklad:

Připojení 8255 pro rozšíření I/O prostoru k zapojení podle **obr. 1.** Programování je zřejmé při volbě instr. MOVX @ Ro,A. Nesmí se v tomto případě použít externí paměť dat, není-li blokována programovým způsobem přes tento port.

Popis SW (pro 8035 – obr. 1)

Základní programové vybavení je tvořeno tak, že se dá přizpůsobit jednoduchou změnou inicializačních proměnných k příslušnému zapojení.

Jádrem programového vybavení je subrutina „REFRSH", která je vyvolána přerušením od časovače a zabezpečuje obsluhu displeje v multiplexním provozu kopírováním registrů displeje z paměti RAM mikropočítače. Zároveň strobuje klávesnici. Vždy po jednom volání způsobí zobrazení 1 znaku a strobování 1 řádku klávesnice. Osmibitový datový vzorek v paměti RAM představuje příslušný znak s desetinnou tečkou ve tvaru (PGFEDCBA), kde (P) je desetinná tečka a (A–G) jsou buzené segmenty displeje. Tato rutina používá banku Rb1 a proměnné LASTKY, CURDIG a F1 dávají status z této subrutiny. Zobrazování probíhá automaticky přerušováním od časovače zprava doleva až po délku displeje danou NEXTPL.

Základní použití rutin

Funkce jednotlivých rutin:

KBDIN – Vstup z klávesnice – čeká dokud nebyl přijat z klávesnice jeden tlačítkový vstup a vrací se s dekódovanou hodnotou ve střádači.

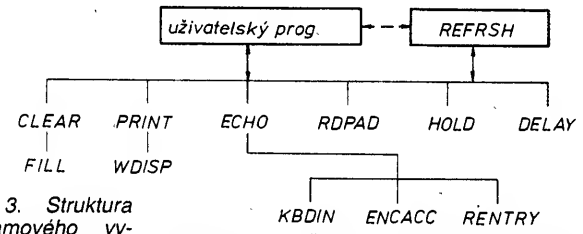
CLEAR – Nuluje displej.

ENCACC – Dekóduje binární vzorek ve střádači ve vztahu k zobrazení na displeji. Používá ACCU pro přístup k dekódovací tabulce.

WDISP – Zápis na displej; zapisuje obsah střádače do další pozice na displeji. Při opakovaném volání automaticky posouvá zobrazení na displeji.

RENTRY – Pravostranný výstup – ukládá obsah střádače do pravé krajní pozice, přitom ostatní znaky posouvá doleva.

PRINT – Zapisuje pevně uživatelsky definované zprávy ukončené kódem 00H a uložené v paměti programu (ROM) na displej.



Obr. 3. Struktura programového vybavení (922–3)

ECHO – Čeká na stisknutí klávesy a její význam vypíše na displej. Vhodná pro kontrolu dat zadávaných z klávesnice.

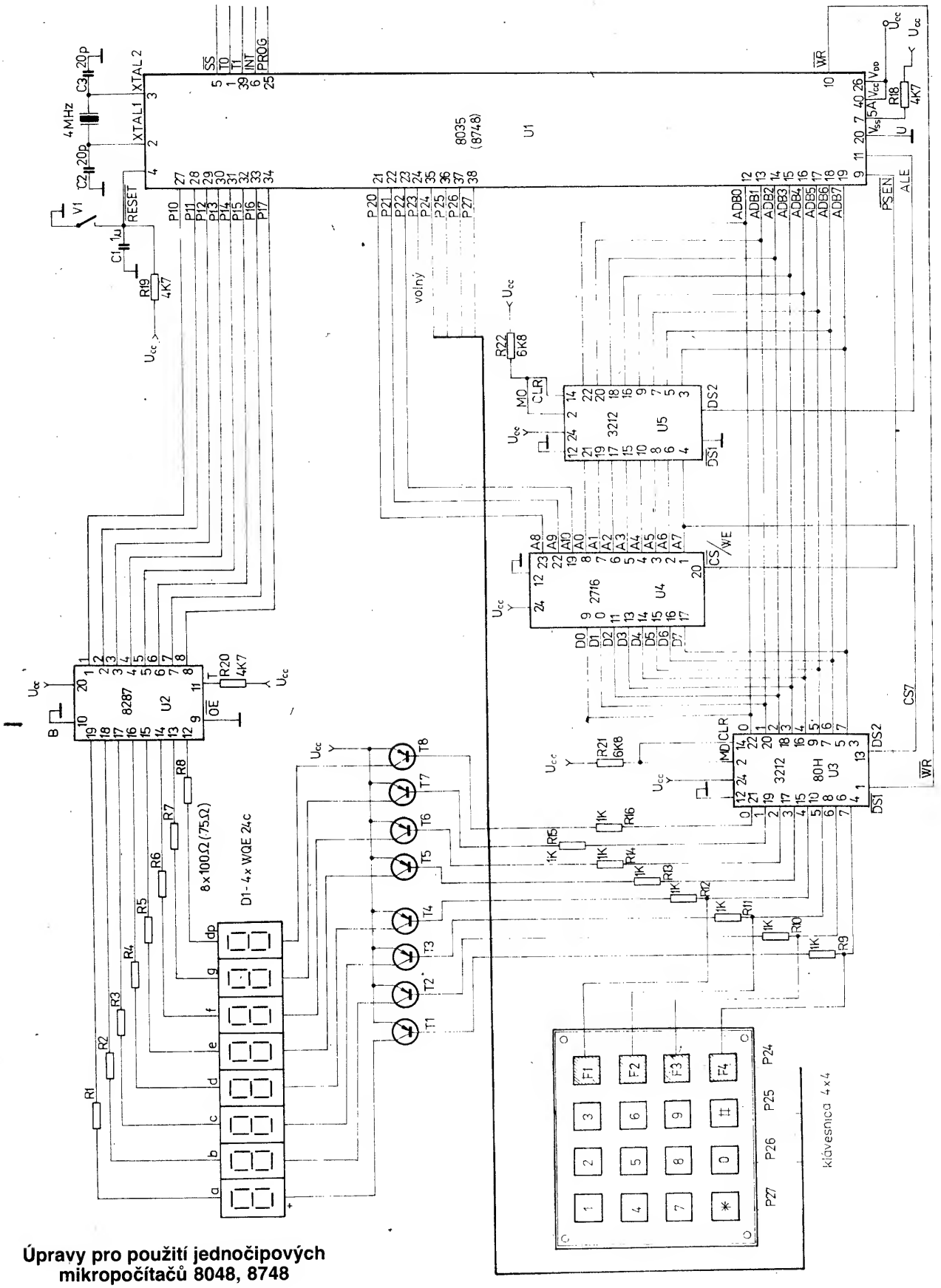
RDPADD – Přidá nebo vymaže desetinnou tečku za zadaným číslem. Vhodná pro zadávání desetinných čísel.

HOLD – Je vyvolána po dobu stisknutí klávesy. Vhodná pro autoinkrementy nebo dekrementy v závislosti na stavu klávesnice.

DELAY – Podprogram pro generování programového zpoždění v závislosti na počtu cyklů REFRESH a obsahu střádače před voláním. Vhodný pro blikání displeje a jiné jednoduché časové programové operace.

Algoritmus pro snímání klávesnice je utvořen tak, že při stisku klávesy musí proběhnout definovaný počet snímacích cyklů k potvrzení skutečného stisknutí. Je možné vyhodnotit stisknutí definované klávesy s funkcí SHIFT nebo CONTROL. Je-li stisknutá klávesa dekódována a potvrzena, je její pozice vyjádřena číselnou hodnotou. Tento kód je uložen do paměti na adresu KBDBUF. Dekódovací rutina potom čte obsah na této adrese ke zjištění, která klávesa byla stisknuta.

Výpis programu je dosti komentován. Význam jednotlivých proměnných, částí programu a vazeb je z komentáře patrný. Závěrem je uveden krátký testovací program vyvolaný funkčními tlačítky 1 až 4. Tento program (resp. programy) začínají na ORG 11Fh.



Obr. 1. Zapojení klávesnice a displeje s 8035 (922–1)

Úpravy pro použití jednočipových mikropočítačů 8048, 8748

Zapojení

Jak je zřejmé z **obr. 5,** je sběrnice použita jako výstupní port pro buzení číslic (DIGIT) v případě, že v aplikacích vystačíme s vnitřními možnostmi mikropočítače. Části portu P2 můžeme použít jako expanderové připojovací vodiče nebo jako CS pro připojení dalších I/O obvodů na sběrnici. Sběrnici a P1 je vhodné připojit na +5 V přes rezistory 10 kΩ.

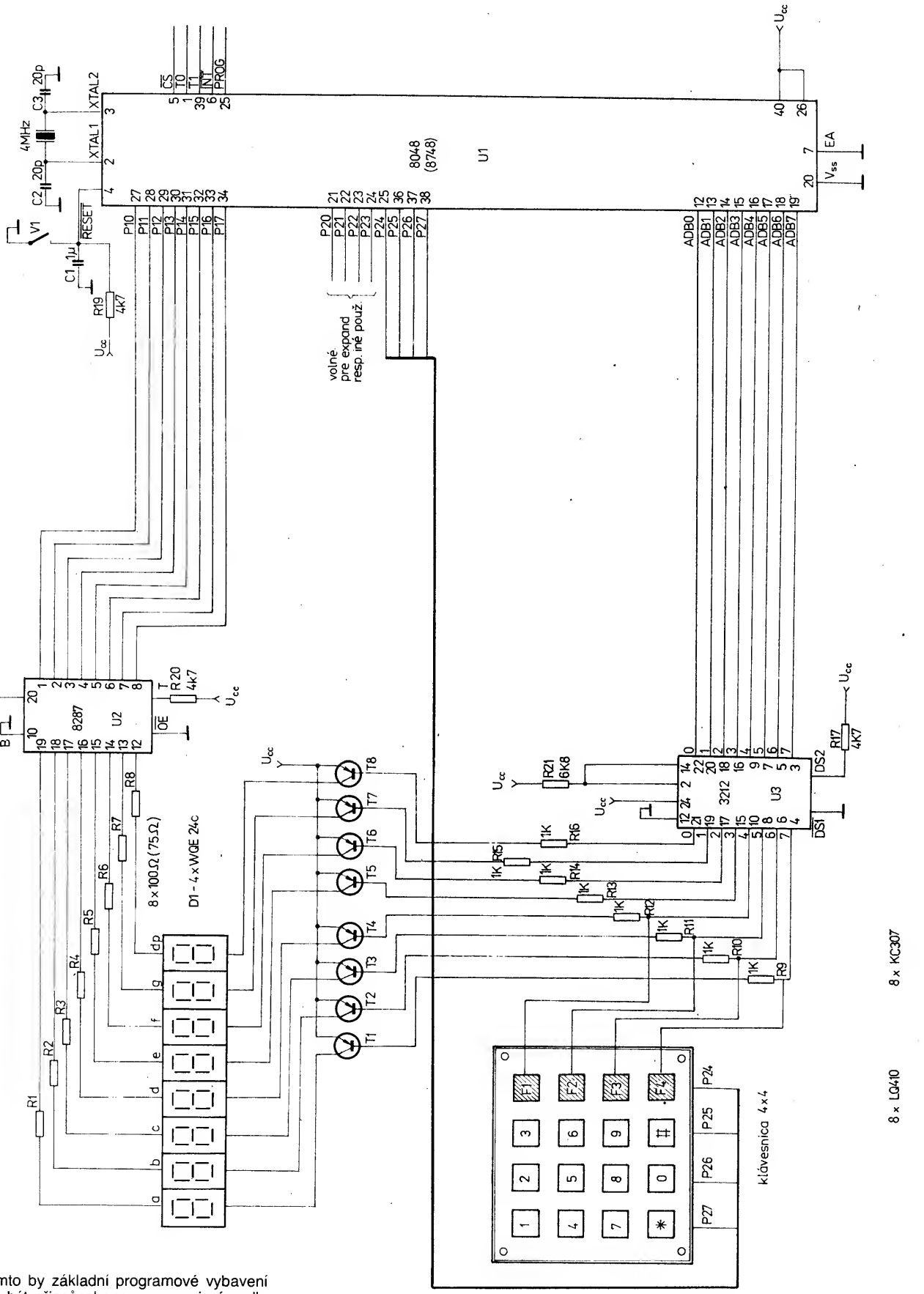
Programová obsluha

Program se použije stejný, jako pro 8035, jen se vypustí v definiční oblasti registr banky 0 ACCSA a ROSA (uvolníme si registry R4 a R3 pro použití). Zařadí se instrukce OUTL PDIGIT,A (označena # #) v řádku 106 a vypustí se instrukce:

```
SEL RB0
MOV ACCSA,A
MOV A,R0
MOV R0SA,A
MOV R0,#80H
MOV A,ACCSA
```

```
MOV X @RQ,A
MOV A,R0SA
MOV R0,A
MOV A,ACCSA
SEL RB1
```

(řádky 107 až 117).



Obr. 5. Zapojení klávesnice a displeje s 8048 (8748) (922–5)

Tímto by základní programové vybavení mělo být přizpůsobeno pro zapojení podle **obr. 2.**

Závěr

Předložené zapojení tvoří dobrý základ celé řadě aplikací, kde se vyskytuje použití klávesničky a displeje. Mohou to být regulátory, ovládací pulty, hračky, jednoduché autonomní řídicí členy a podobně.

Program podle **výpisu 1** s použitím zapojení podle **obr. 1** (8035) je odladěný a funkční klávesy 1 až 4 způsobí:

F1 – žobrazí na displejí číslice od 8 do 1,
F2 – pozdraví vás „AHOJ",
F3 – vypíše na displej pomlčky „–",
F4 – při zadávání číslic vloží desetinnou tečku.

Literatura

Nohel, Štětinová: Assembler 48.
Barták: Mikrořadiče 48.
Intel: 8048 Family applications handbook.

```
                        1 ;************************************************************************
                        2 ;
                        3 ;                 INICIALIZACNI PRIRAOENI PRO OEFINICI KONFIGURACE
                        4 ;
                        5 ;************************************************************************
                        6 ;
0010                    7 PDIGIT  EQU             8U8     ;8UDI CISLICE A STROBUJE RAOKY KLAVESNICE
0008                    8 PSGMNT  EQU             P1      ;8UDI SEGMENTY CISLIC OISPLAY V ZAVISLOSTI OO BUZENE CISLICE
0009                    9 PINPUT  EQU             P2      ;SNIMA STLACENOU KLAVESU BUZENEHO RAOKU
                       10 ;
0000                   11 POSLOG  EQU             00H ;POSITIVNI LOGIKA POUZITYCH PRVKU
00FF                   12 NEGLOG  EQU         0FFH ;NEGATIVNI LOGIKA POUZITYCH PRVKU
                       13 ;
00FF                   14 CHRPOL  EQU             NEGLOG  ;OEF,ZOA VYSTUPNI VODICE JSOU AKTIVNI HI=LO=PDIGIT
0000                   15 SEGPOL  EQU             POSLOG  ;OEF,ZOA VYSTUPNI VOOICE JSOU AKTIVNI HI=LO=PSGMNT
00F0                   16 INPMSK  EQU         0F0H        ;DEF,BITY POUZITE JAKO VSTUPNI
                       17 ;
0008                   18 CHARNO  EQU             8               ;POCET CISLIC OISPLAY (POCET SEDMISEGMENTOVEK)
0004                   19 NROWS   EQU             4               ;POCET RAOKU KLAVESNICE(MENSI NEBO = CHARNO)
0004                   20 NCOLS   EQU             4               ;MENSI OIMENZE KLAVESNICOVE MATICE
                       21 ;
FFF0                   22 TICK    EQU             -10H    ;OETERMINUJE INTERUPT INTERVAL FEFRSH RUTINY
0004                   23 OE8NCE  EQU             4               ;POCET USPESNYCH SNIMANI KLAVESY KU JEJI OETEKCI
0000                   24 BLANK   EQU             00H             ;KOO MEZERY OISPLAY (BLANK)
                       25 ;
000F                   26 ENCMSK  EQU             0FH             ;VYBER KTERE BITY JSOU PLATNE PRI CTENI Z PORTU P2
                       27 ;
                       28 ;************************************************************************
                       29 ;
                       30 ;               POUZITE REGISTRY BANKY 0
                       31 ;
                       32 ;POINTERY PRO AORESACI PAMETI + USCHOVA ACCU A R0 PRI POUZITI 8035
0000                   33 PNTR0   EQU             R0
0001                   34 PNTR1   EQU             R1
0003                   35 ACCSA   EQU             R3              ;POUZITO PRO USCHOVU ACCU PRI 8035
0004                   36 R0SA    EQU             R4              ;POUZITO PRO USCHOVU R0 PRI 8035
0007                   37 NEXTPL  EQU             R7              ;UKAZATEL NASLEONE OISPL.POSICE PRO ZAPIS
                       38 ;
                       39 ;************************************************************************
                       40 ;
                       41 ;               POUZITE REGISTRE BANKY 1
                       42 ;
                       43 ;PNTR0   EQU             R0              ;(UZ BYLO OEFINOVANO)
                       44 ;PNTR1   EQU             R1
0002                   45 ASAVE   EQU             R2              ;ORZI HODNOTU ACCU BEHEM REFRESH RUTINY
0004                   46 ROTPAT  EQU             R4              ;ORZI VSTUPNI VZOREK BEHEM ROTACE PRES CY
0005                   47 ROTCNT  EQU             R5              ;POCITA BITY PROCHAZEJICI PRES CY
0006                   48 LASTKY  EQU             R6              ;ORZI POSICI POSLEONI STLACENE KLAVESY
0007                   49 CURDIG  EQU             R7              ;ORZI POSICI PRISTIHO OISPLAY=ZOBRAZ.ZNAKU
                       50 ;
                       51 ;************************************************************************
                       52 ;
                       53 ;               OATA RAM ALOKACE
                       54 ;
0020                   55 NREPTS  EQU             32              ;ORZI POCET USPESNYCH SEJMUTI OETEKOVANE KLAVESY
0021                   56 KEYLOC  EQU             33              ;INKREMENTOVANO PRI USPESNEM SEJMUTI KLAVESY
0022                   57 KBOBUF  EQU             34              ;PRENASI POSICI STLACENI KLAVESY Z REFRESH RUTINY
                       58                                                 ;OO OSTATNICH SUBROUTIN
0023                   59 ROELAY  EQU             35              ;NENULOVE PRI FUNKCI REFRESH RUTINY
                       60 ;
                       61 ;
0037                   62 SEGMAP  EQU             (63-CHARNO)     ;BASE OISPLAY REGISTRU V INTERNI RAM
                       63 ;                                ; MUZE BYT KOEKOLIV V INTERNI PAMETI RAM
                       64 ;
                       65 ;************************************************************************
                       66 ;
                       67 ;========================================================================
                       68 ;
0000                   69 ORG             00H
0000 0468              70                 JMP             INIT    ;SKOK NA INICIALIZACNI PROCEOURU
                       71 ;
                       72 ;
0007                   73 ORG             07H
                       74 ;
                       75 ;TIINT  TIMER INTERUPT SUBROUTINA
                       76 ;               CALL AOR 007H PRI PRETECENI CASOVACA
                       77 ;               TIMER TAOY MUZE BYT PREINICIALIZOVANY PODLE POTREBY UZIVATELE
                       78 ;               ZDE SE PROVEOE OSVEZENI OISPLAY A SNIMANI KLAVESNICE
                       79 ;               VZDY PO JEONE POSICI PRI KAZOEM VOLANI
0007 D5                80 TIINT:  SEL             RB1
0008 AA                81                 MOV             ASAVE,A
0009 23F0              82                 MOV             A,#TICK
0008 62                83                 MOV             T,A                     ;REINICIALIZACE TIMERU
                       84 ;
                       85 ;************************************************************************
                       86 ;
                       87 ;               VLASTNI UZIVATELSKA INTERUPT RUTINA MUZE BYT ZDE UMISTENA
                       88 ;               PRO LIBOVOLNE POUZITI VE VAZBE NA PRERUSENI CASOVACE
                       89 ;
                       90 ;************************************************************************
000C 1410              91                 CALL    REFRSH          ;VOLANI OISPLAY/KLAVESNICOVE SUBROUTINY
                       92 ;TIRET  OBNOVA ACCU PREO NAVRATEM Z REFRESH SUBROUTINY
000E FA                93 TIRET:  MOV             A,ASAVE
000F 93                94                 RETR
                       95 ;************************************************************************
                       96 ;
                       97 ;REFRSH SUBROUTINA MULTIPLEXUJE 7 SEG.OISPLAY A VZOY ZOBRAZI OALSI POSICI
                       98 ;NA OISPLAY.JE VHOONE JI VOLAT V MSEC.INTERVALECH
                       99 ;************************************************************************
                      100 ;
0010 2300             101 REFRSH: MOV             A,#BLANK XOR SEGPOL
0012 39               102                OUTL    PSGMNT,A        ;ZAPIS BLANK KODU OO PSGMNT
0013 2362             103 REFR1:  MOV             A,#CHRST8
0015 6F               104                ADO             A,CUROIG
0016 A3               105                MOVP    A,@A
                      106 ;##            OUTL    POIGIT,A        ;BUZENI ZNAKU PRI POUZITI 8748
0017 C5               107                SEL             RB0                     ;R0 BANK VYBER JEN PRO 8035
0018 AB               108                MOV             ACCSA,A         ;USCHOVA ACCU DO ACCSA JEN PRO 8035
0019 F8               109                MOV             A,R0            ;USCHOVA R0 OO R0SA JEN PRO 8035
```

```
001A AC          110                  MOV             R0SA,A          ;PRO 8035
001B B880        111                  MOV             R0,#080H        ;BIT 7 SET JAKO ADRESA 3212 PRO INSTR,MOVX @R0,A
001D FB          112                  MOV             A,ACCSA         ;OBNOVA ACCU PRO 8035
001E 90          113                  MOVX    @R0,A           ;ZAPIS DO OIGIT LATCH (3212) JEN PRO 8035
001F FC          114                  MOV             A,R0SA          ;OBNOVA R0 JEN PRO 8035
0020 A8          115                  MOV             R0,A            ;PRO 8035
0021 FB          116                  MOV             A,ACCSA         ;OBNOVA ACCU JEN PRO 8035
0022 D5          117                  SEL             RB1                     ;VYBER RB1 PRO 8035
                 118 ;
                 119 ;  PRI POUZITI 8748 SE INSTR.OO SEL RB0 PO SEL RB1 VYPOUSTEJI (JEN PRO 8035)
                 120 ;=============================================================================
0023 2337        121          MOV             A,#SEGMAP       ;DEFINICE BAZE DISPLAY-REGISTRU V RAM
0025 6F          122                  ADD             A,CURDIG        ;ADD CURDIG POSUNUTI
0026 A9          123                  MOV             PNTR1,A
0027 F1          124                  MOV             A,@PNTR1        ;NAHRAVANI VZORKY
0028 39          125                  OUTL    PSGMNT,A        ;BUDI PRISLUSNE SEGMENTY
                 126 ;
                 127 ;*************************************************************
                 128 ;               DALSI ZNAK JE NYNI ZOBRAZEN NA DISPLAY
                 129 ;               KLAVESNICOVA SNIMACI RUTINA JE ZARAZENA DO DISPLAY PROCEDURY
                 130 ;               S VYUZITIM BUZENI PRISLUSNE CISLICE A TIM I PRISL.RADKU KLAV.
                 131 ;*************************************************************
                 132 ;
0029 B821        133 SCAN:   MOV              PNTR0,#KEYLOC   ;NASTAVUJE POINTER NA UKAZATEL POCTU SNIMANI.
002B 0A          134                  IN              A,PINPUT        ;NACITA STLACENI KLAVESY
                 135 ;
                 136 ;*************************************************************************
                 137 ;
                 138 ;*************************************************************
                 139 ;               ROTACE ACCU PRES CY
                 140 ;*************************************************************
002C BD04        141 SCAN1:  MOV              ROTCNT,#NCOLS   ;NASTAVUJE ROTACNI SMYCKU POCLE POCTU SLOUPCU
002E F7          142 NXTLOC: RLC              A
002F AC          143                  MOV             ROTPAT,A        ;UCHOVAVA POSUNUTOU VZORKU
0030 F64A        144                  JC              SCAN5           ;JEDNOTKOVY BIT INDIKUJE ZE NEBYLA STLACENA KL.
                 145 ;
                 146 ;*************************************************************
                 147 ;V TOMTO BODE PROGRAMU BYLO PRAVE ZJISTENO ZE HODNOTA V KEYLOC JE POSICE
                 148 ;TLACITKA KTERE JE NYNI STLACENE.NASLEDUJICI INSTRUKCE OEKODUJI A URCUJI
                 149 ;VYZNAM ZMACKNUTEHO TLACITKA.JAK POUZIVAME MODIFIKACNI VYZNAM FUNKCE
                 150 ;TLACITKA(SHIFT,CONTROL)NEBO JINY MODALNI VYZNAM TLACITEK MELO BY SE TO
                 151 ;PROVADET V TOMTO BODE PROGRAMU,PREDTIM NEZ ZACNE DALSI DEKODOVACI LOGIKA
                 152 ;NAPRIKLAD V TETO CASTI MUZE BYT KEYLOC=PROMENNA POROVNAVANA S NAMI
                 153 ;URCENOU POSICI MODALNIHO TLACITKA A PRI JEHO IDENTIFIKACI SI NASTAVIME
                 154 ;NEJAKY FLAG S SKOCIME NA NAV."SCAN5"A NEBO POROVNAVAT PROM.KEYLOC S
                 155 ;PREDESLYM JEJIM OBSAHEM
                 156 ;*************************************************************
0032 A5          157                  CLR             F1              ;ZNACKA ZE NEJMENE JEDNA KLAVESA BYLA DETEKOVANA
0033 B5          158                  CPL             F1              ;NEGOVANO V BEZNEM SNIMANI
                 159 ;
                 160 ;*************************************************************
                 161 ;               ZMACKNUTI TLACITKA BYLO DETEKOVANO V NEKTEREM SLOUPCI JEHO
                 162 ;               JEHO POSICE JE V KEYLOC ZJISTUJE SE JESTLI TO SAME BYLO
                 163 ;               STLACENE V MINULEM CYKLU
                 164 ;*************************************************************
                 165 ;
0034 F0          166                  MOV             A,@PNTR0                ;PNTR0 STALE UKAZI #KEYLOC
0035 2E          167                  XCH             A,LASTKY
0036 DE          168                  XRL             A,LASTKY
0037 B820        169                  MOV             PNTR0,#NREPTS   ;PRIPRAVA NA KONTROLU
                 170                                                          ;OPAKOVACI CITAC
0039 C63F        171                  JZ              SCAN3
                 172 ;
                 173 ;*************************************************************
                 174 ;               JINE TLACITKO BYLO PRECTENO V TOMTO CYKLE NEZ PREDTIM
                 175 ;               NASTAVUJE NREPTS NA POZADOVANY PARAMETR PRO NOVE COUNTDOWN
                 176 ;*************************************************************************
                 177 ;
003B B004        178                  MOV             @PNTR0,#DEBNCE
003D 044A        179                  JMP             SCAN5
                 180 ;
                 181 ;*************************************************************
                 182 ;               TO SAME TLACITKO BYLO DETEKOVANO JAKO V PREDESLEM CYKLU
                 183 ;               PODIVEJ SE NA NREPTS JESTLI JE NULOVE,NEDELEJ NIC
                 184 ;               JESTE DEKREMENTUJ NREPTS
                 185 ;               JE LI NULA PRESUN LASTKY DO KBDBUF
                 186 ;*************************************************************
                 187 ;
003F F0          188 SCAN3:  MOV              A,@PNTR0
0040 C64A        189                  JZ              SCAN5           ; JE LI NULA
0042 07          190                  DEC             A               ;INDIKUJE JEDNO NAVIC USPESNE TLAC.ZMACKNUTI
0043 A0          191                  MOV             @PNTR0,A
0044 964A        192                  JNZ             SCAN5           ;JAK VYSLEDEK DEKREMENTACE JE NENULOVY
0046 FE          193                  MOV     A,LASTKY
0047 B822        194                  MOV             PNTR0,#KBDBUF
0049 A0          195                  MOV             @PNTR0,A        ;OZNACUJE NOVE STLACENI TLACITKA
                 196 ;
004A B821        197 SCAN5:  MOV      PNTR0,#KEYLOC
004C 10          198                  INC             @PNTR0
004D FC          199                  MOV             A,ROTPAT
004E E02E        200                  DJNZ    ROTCNT,NXTLOC
                 201 ;
0050 EF62        202 SCAN6:  DJNZ     CURDIG,SCAN9
                 203 ;
                 204 ;*************************************************************
                 205 ;               NASLEDUJICI INSTRUKCE JSOU POUZIVANY KLAVESNICOVOU RUTINOU
                 206 ;               (SCANING)A JSOU VYKONAVANY POUZE PO REFRESH CELEHO DISPLAY
                 207 ;               TEDY VSECH CISEL DANYCH CHARNO
                 208 ;*************************************************************
                 209 ;
0052 BF08        210                  MOV             CURDIG,#CHARNO
0054 B000        211                  MOV             @PNTR0,#0               ;PNTR0 STALE OBSAHUJE #KEYLOC
0056 765A        212                  JF1             SCAN8           ;JUMP JAK NEJAKE TLACITKO BYLO DETEKOVANO
0058 BEFF        213                  MOV             LASTKY,#0FFH    ;MENI (LASTKY) JEN KDYZ BYLO STLACENO ZADNE TL.
005A A5          214 SCAN8:  CLR              F1
                 215 ;
                 216 ;*************************************************************
                 217 ;               DALSI PROGRAMOVY SEGMENT JE INTERUPT-RIZENA CAST "DELAY"
                 218 ;               POUZIVAJIC DEKREMENTACI RAM LOKACE "RDELAY" 1=KRAT ZA DISPL.
                 219 ;               REFRESH DOKUD "RDELAY" NENI NULA
```

```
                    220 ;*****************************************************************
                    221 ;
0058 B923           222                 MOV             PNTR1,#RDELAY
005D F1             223                 MOV             A,@PNTR1
005E C662           224                 JZ              SCAN9
0060 07             225                 DEC             A
0061 A1             226                 MOV             @PNTR1,A
0062 83             227 SCAN9:  RET
                    228 ;
                    229 ;*****************************************************************
                    230 ;
                    231 ;CHRSTB JE BAZE VRCHOLU BINARNICH VZOREK POVOLUJICI SVIT 1=SEOMISEGMENTOVKY
0062                232 CHRSTB  EQU             ($-1) AND 0FFH
0063 FE             233                 DB              (00000001B XOR CHRPOL)
0064 FD             234                 DB              (00000010B XOR CHRPOL)
0065 FB             235                 DB              (00000100B XOR CHRPOL)
0066 F7             236                 DB              (00001000B XOR CHRPOL)
0067 EF             237                 DB              (00010000B XOR CHRPOL)
0068 DF             238                 DB              (00100000B XOR CHRPOL)
0069 BF             239                 DB              (01000000B XOR CHRPOL)
006A 7F             240                 DB              (10000000B XOR CHRPOL)
                    241 ;
                    242 ;INIT   INITIALIZACE PROCESOROVYCH REGISTRU
006B D5             243 INIT:   SEL             RB1
006C BF08           244                 MOV             CUROIG,#CHARNO
006E B822           245                 MOV             PNTR0,#KBDBUF
0070 B0FF           246                 MOV             @PNTR0,#0FFH
0072 B821           247                 MOV             PNTR0,#KEYLOC
0074 B000           248                 MOV             @PNTR0,#0
0076 23F0           249                 MOV             A,#INPMSK
0078 3A             250                 OUTL    PINPUT,A                ;NASTAVUJE PINPUT NA LOG.1 PRO STAHOVANI DO 0
0079 C5             251                 SEL             RB0
007A 14A9           252                 CALL    CLEAR                   ;INICIALIZACE DISPLAY REGISTRU V RAM
007C A5             253                 CLR             F1
007D 23F0           254                 MOV             A,#TICK                 ;NASTAVENI INTERUPT INTERVALU
007F 62             255                 MOV             T,A
0080 55             256                 STRT    T
0081 25             257                 EN              TCNTI                   ;POVOLUJE TIMER INTERUPT
                    258 ;
                    259 ;
                    260 ;*****************************************************************
                    261 ;
                    262 ;ECHO   KONTROLA ZDA NEJAKE NOVE STLACENI TLAC.BYLO DETEKOVANO
                    263 ;               TRANSFORMUJE KAZDE STLACENI DO DISPILSEGMENTOVE VZORKY
                    264 ;               A ZAPISUJE TO DO PRISLUSNEHO DISPLAY REGISTRU V RAM
                    265 ;*****************************************************************
0082 148E           266 ECHO:   CALL    KBDIN                   ;ZJISTENI NOVEHO STLACENI
0084 B28C           267                 JB5             FKEY            ;SKOK JE LI DETEKOVANO TL.V PRAVEM SLOUPCI
                    268 ;               PROTOZE ACCU JE POUZIVANY ENCACC A RENTRY RUTINAMI JEHO OBSAH
                    269 ;               MUSI BYT ZPRACOVANY NEBO USCHOVANY PRED VOLANIM ENCACC
0086 14C5           270                 CALL    ENCACC                  ;TVORI PRISLUSNOU VZORKU SEGMENT ZOBRAZENI
0088 14E6           271                 CALL    RENTRY                  ;ZAPISUJE VZORKU DO DISPLAY REGISTRU V RAM
008A 0482           272                 JMP             ECHO                    ;SMYCKA
                    273 ;
008C 241F           274 FKEY:   JMP             FUNCTN                  ;JUMP MIMO STRANKU (OF PAGE CODE) DO DEMONSTR,
                    275                                                             ;RUTIN RESP.UZIVATELSKY DEFINOVANYCH
                         RUTIN
                    276 ;*****************************************************************
                    277 ;               IMPLEMENTACE NASLEDUJICICH SUBROUTIN SE SPOLECNE POUZIVA
                    278 ;               PRO VETSINU KLAVESNICOVE/DISPLAY ORIENTOVANYCH APLIKACI
                    279 ;               MOHOU BYT POUZITE PRESNE JAK JSOU UVEDENE NEBO PRISPUSOBENE
                    280 ;               PRO SPECIALNI POUZITI
                    281 ;*****************************************************************
                    282 ;KBOIN  KLAVESNICOVA VSTUPNI SUBROUTINA
                    283 ;               MUZE BYT POUZITA KU SPOJENI UZIVATELSKEHO PROGRAMOVEHO POZADI
                    284 ;               S PRERUSENIM RIZENOU KLAVESNICOVOU SUBROUTINOU (SCANER)
                    285 ;               DAVA POUZE PO STLACENI NOVEHO TLACITKA DEKODOVANOU HODNOTU
                    286 ;               (LEPE NEZ JEHO POSICI V MATICI TLACITEK KLAVESNICE) V ACCU
                    287 ;
008E B922           288 KBOIN:  MOV             PNTR1,#KBDBUF
0090 2380           289                 MOV             A,#80H                  ;KBDBUF BUDE MASKOVAN JAKO CISTY
0092 21             290                 XCH             A,@PNTR1                ;NAPLNENI BUFFROVANOU HODNOTOU
0093 F28E           291                 JB7             KBOIN
0095 0399           292                 ADD             A,#LEGNDS       ;PRICTENI BASE DEKODOVACI TABULKY TLACITEK
0097 A3             293                 MOVP    A,@A                    ;OBDRZENI PRISLUSNEHO VYZNAMU TLACITKA
0098 83             294                 RET
                    295 ;
                    296 ;
                    297 ;LEGNDS JE BAZE TABULKY UKAZUJICI VYZNAM TLACITEK MATICE KLAVESNICE
                    298 ;               PRO KLAVESNICI POUZITOU V TETO UKAZCE APLIKACE
                    299 ;               S USPORADANIM PODLE OBRAZKU OBR.1
                    300 ;           BITY 6-4 MOHOU BYT POUZITY POUZITY K DEKODOVANI TYPU TLACITEK
                    301 ;               S TIMTO VYZNAMEM;
                    302 ;                                       BIT4 INDIKUJE SKUTECNE DEKADICKE CISLO
                    303 ;                                       BIT5 INDIKUJE FUNKCNI TLACITKA V PRAVEM SLOUPCI KLAV.
                    304 ;                                       BIT6 INDIKUJE FUNKCNI ZNAKY (* A #)
0099                305 LEGNDS  EQU             ($ AND 0FFH)    ;POUZIVA SPODNI BITY JAKO TABULKOVY INDEX
0099 4F             306                 DB              4FH
009A 10             307                 DB              10H
009B 4E             308                 DB              4EH
009C 28             309                 DB              28H     ; PDIGIT4==>    1       2       3       <1>
009D 17             310                 DB              17H
009E 18             311                 DB              18H     ; PDIGIT5==>    4       5       6       <2>
009F 19             312                 DB              19H
00A0 24             313                 DB              24H     ; PGIGIT6==>    7       8       9       <3>
00A1 14             314                 DB              14H
00A2 15             315                 DB              15H     ; PDIGIT7==>    *       0       #       <4>
00A3 16             316                 DB              16H
00A4 22             317                 DB              22H     ;               |       |       |       |
00A5 11             318                 DB              11H     ;               |       |       |       |
00A6 12             319                 DB              12H     ;               V       V       V       V
00A7 13             320                 DB              13H     ;           PINPUT7 PINPUT6 PINPUT5 PINPUT4
00A8 21             321                 DB              21H
                    322 ;
                    323 ;*****************************************************************
                    324 ;CLEAR  ZAPISUJE 'BLANK' ZNAK DO DISPLAY REGISTRU V RAM
                    325 ;               VRACI SE S NASTAVENYM NEXTPL NA LEVOU KRAJNI POSICI DISPLAY
                    326 ;FILL   ZAPISUJE SEGMENTOVOU VZORKU V ACCU DO DISPLAY REGISTRU V RAM
00A9 2300           327 CLEAR:  MOV             A,#BLANK XOR SEGPOL
00AB B938           328 FILL:   MOV             PNTR1,#SEGMAP+1
```

```
00A0 BF08          329                    MOV             NEXTPL,#CHARNO
00AF A1            330 CLR1:   MOV                 @PNTR1,A                        ;UKLADA BLANK KOD
00B0 19            331                    INC             PNTR1                           ;DALSI POSICE DO LEVA
00B1 EFAF          332                    DJNZ    NEXTPL,CLR1
00B3 BF08          333                    MOV             NEXTPL,#CHARNO
00B5 83            334                    RET
                   335 ;
                   336 ;*************************************************************************
                   337 ;
                   338 ;PRINT  SUBROUTINA KE KOPIROVANI BINARNICH VZOREK Z "ROM" DO
                   339 ;               DISPLAY REGISTRU V RAM POCINAJE ADRESOU V PNTR0
                   340 ;               DOKUD SE NEDOSAHNE ZAKONCOVACI KOD (V TOMTO PRIPADE =00H=)
                   341 ;               !!! RETEZEC ZNAKU MUSI BYT NA TEZE STRANCE JAKO SUBROUTINA
                   342 ;               PRINT Z DUVODU VAZBY NA VOLANI "WDISP NEBO RENTRY"
                   343 ;               PRO EFEKTNI ZAPISY DO DISPLAY REGISTRU V RAM
00B6 F8            344 PRINT:  MOV             A,PNTR0                         ;ZADAVANI NASLED.ZNAKOVE POSICE
00B7 A3            345                    MOVP    A,@A                            ;NACTENI BITOVE VZORKY
00B8 C601          346                    JZ              PNTR1                           ;DEKODOVANI ZAVERECNEHO PRVKU (00H)
00BA 14DB          347                    CALL    WOISP                           ;VYSTUP NA DALSI ZNAKOVOU POSICI
                   348 ;##                CALL    RENTRY                          ;NEBO SKOK NA RENTRY
                   349 ;
00BC 18            350                    INC             PNTR0                           ;INDEX UKAZOVATEL
00BD 0486          351                    JMP             PRINT
00BF 83            352 PRNT1:  RET                                                 ;PROVEDENO
                   353 ;
                   354 ;*************************************************************************
                   355 ;
                   356 ;AHOJ   OBLAST PAMETI ROM S POZDRAVNOU ZPRAVOU 'AHOJ' ( !TEST2')
                   357 ;               (UPOZORNENI ZE POUZE "HOJ" SE VYPISE V PRIPADE RENTRY)
00C0               358 AHOJ    EQU             $ AND OFFH
00C0 77            359                    DB              01110111B XOR SEGPOL
00C1 76            360                    DB              01110110B XOR SEGPOL
00C2 3F            361                    DB              00111111B XOR SEGPOL
00C3 1E            362                    DB              00011110B XOR SEGPOL
00C4 00            363                    DB              00
                   364 ;
                   365 ;*************************************************************************
                   366 ;
                   367 ;ENCACC DEKODUJE LSNIBLE(SPODNI CAST ACCU)NA HEXA KOD
00C5 530F          368 ENCACC: ANL             A,#ENCMSK
00C7 03CB          369                    ADD             A,#DGPATS
00C9 A3            370                    MOVP    A,@A
00CA 83            371                    RET
                   372 ;DGPATS JE BAZE TABULKY SEGMENTOVYCH VZOREK PRO ZAKLADNI
                   373 ;ZNAKY (0-F)
                   374 ;PRO SPECIFICKE UZIVATELSKE APLIKACE TABULKA MUZE BYT MODIFIKOVANA
                   375 ;PODLE POTREBY  S POUZITIM FORMATU SEGMENTU DISPLAY
                   376 ;FORMAT JE V NASEM PRIPADE STANDARTNI--P-G-F-E-D-C-B-A--
                   377 ;KDE -P-JE SVIT DESETINNE TECKY
                   378 ;
00CB               379 DGPATS  EQU             $ AND OFFH
00CB 3F            380                    DB              00111111B XOR SEGPOL
00CC 06            381                    DB              00000110B XOR SEGPOL
00CD 5B            382                    DB              01011011B XOR SEGPOL
00CE 4F            383                    DB              01001111B XOR SEGPOL
00CF 66            384                    DB              01100110B XOR SEGPOL
00D0 6D            385                    DB              01101101B XOR SEGPOL
0001 7D            386                    DB              01111101B XOR SEGPOL
00D2 07            387                    DB              00000111B XOR SEGPOL
00D3 7F            388                    DB              01111111B XOR SEGPOL
00D4 67            389                    DB              01100111B XOR SEGPOL
0005 77            390                    DB              01110111B XOR SEGPOL
00D6 7C            391                    DB              01111100B XOR SEGPOL
00D7 39            392                    DB              00111001B XOR SEGPOL
00D8 5E            393                    DB              01011110B XOR SEGPOL
0009 79            394                    DB              01111001B XOR SEGPOL
00DA 71            395                    DB              01110001B XOR SEGPOL
                   396 ;
                   397 ;*************************************************************************
                   398 ;
                   399 ;WDISP  ZAPISUJE BINARNI VZORKU NYNI V ACCU DO DALSI ZNAKOVE POSICE
                   400 ;               DISPLAY (NEXTPL)
                   401 ;
00DB A9            402 WDISP:  MOV             PNTR1,A
00DC FF            403                    MOV             A,NEXTPL
00DD 0337          404                    ADD             A,#SEGMAP
00DF 29            405                    XCH             A,PNTR1
00E0 A1            406                    MOV             @PNTR1,A
00E1 EFE5          407                    DJNZ    NEXTPL,WDISP1
00E3 BF08          408                    MOV             NEXTPL,#CHARND
00E5 83            409 WDISP1: RET
                   410 ;
                   411 ;*************************************************************************
                   412 ;RENTRY SUBROUTINA K ZAPISU ACCU DO PRAVE KRAJNI POSICE DISPLAY
                   413 ;               S POSUNUTIM VSEHO DO LEVA O 1 POSICI
                   414 ;
00E6 B938          415 RENTRY: MOV             PNTR1,#SEGMAP + 1
00E8 BF08          416                    MOV             NEXTPL,#CHARND
00EA 21            417 RENTR1: XCH             A,@PNTR1
00EB 19            418                    INC             PNTR1
00EC EFEA          419                    DJNZ    NEXTPL,RENTR1
00EE BF08          420                    MOV             NEXTPL,#CHARNO          ;OBNOVA UKAZATELE NA LEVOU KRAJNI POSICI

00F0 83            421                    RET
                   422 ;
                   423 ;*************************************************************************
                   424 ;
                   425 ;RDPADO  DAVA DESETINNOU TECKU ZA POSLEDNI ZNAK NA DISPLAY
                   426 ;DPADD   DAVA DESETINNOU TECKU KE ZNAKU KTERY JE V ACCU
                   427 ;
00F1 2301          428 RDPADD: MOV             A,#01H      ;NASTAVENI INDEXU NA PRAVOU KRAJNI POSICI
00F3 0337          429 DPADD:  ADD             A,#SEGMAP       ;SCITA DISPLAY REG.ADRESU PRO PRISL.POSICI
00F5 A9            430                    MOV             PNTR1,A
00F6 F1            431                    MOV             A,@PNTR1
00F7 0380          432                    XRL             A,#80H
00F9 A1            433                    MOV             @PNTR1,A
00FA 83            434                    RET
                   435 ;
                   436 ;*************************************************************************
0100               437                    ORG             0100H   ;POSUV NA DALSI STRANKU PAMETI PROGRAMU
```

```
                    438 ;HOLD   SUBRDUTINA VOLANA KDYZ JE STLACENE NEJAKE TLACITKO
                    439 ;               A NEVRACI SE Z NI OOKUD NENI TOTO UVOLNENO
0100 D5             440 HOLD:   SEL             RB1
0101 FE             441                 MOV             A,LASTKY        ;(LASTKY)=0FFH JAK NENI STLACENO NIC
0102 C5             442                 SEL             RB0
0103 37             443                 CPL             A
0104 9600           444                 JNZ             HOLD
0106 83             445                 RET
                    446 ;
                    447 ;*****************************************************************
                    448 ;
                    449 ;DELAY  SUBROUTINA DAVA SPOZDENI ROVNE POCTU KOMPLETNICH REFRESH OISPLAY
                    450 ;               V ZAVISLOSTI NA HOONOTE ZPOZDENI OANE ACCU PRED JEJIM VOLANIM
0107 B923           451 OELAY:  MOV             PNTR1,#RDELAY
0109 A1             452                 MOV             @PNTR1,A
010A F1             453 OELAY1: MOV             A,@PNTR1
0108 960A           454                 JNZ             OELAY1
010D 83             455                 RET
                    456 ;
                    457 ;*****************************************************************
                    458 ;
011F                459                 ORG             11FH        ;ZACATEK UKAZEK
                    460
                    461 ;PROGRAMY NA TETO STRANCE PAMETI PROGRAMU JSOU UKAZKOU POUZITI
                    462 ;NAVRZENEHO HW A PRISLUSNYCH RUTIN SW
                    463 ;NESLOUZI SAMOZDREJME JAKO STANDARTNI FUNKCE TOHOTO PRIKLADU
                    464 ;APLIKACE JE UKAZKOU OEFINICE UZIVATELSKYCH RUTIN NA ZAKLAOE
                    465 ;VYVOLANI FUNKCNICH TLACITEK
                    466 ;RUTINY JSOU VOLANY JELI 1 ZE 4 F=KEYS STLACENE
                    467 ;
                    468 ;
                    469 ;*****************************************************************
                    470 ;
                    471 ; FUNCTN RUTINY K IMPLEMENTACI 1 ZE 4 OEMONSTRACNICH UKAZEK V ZAVISLOSTI
                    472 ; OD STLACENI 1 ZE 4 FUNKCNICH TLACITEK
                    473 ;
011F 1231           474 FUNCTN: JB0             FUNCT1
0121 3220           475                 JB1             FUNCT2
0123 5229           476                 JB2             FUNCT3
                    477 ;
0125 14F1           478 FUNCT4: CALL    RDPAOD
0127 0482           479                 JMP             ECHD
                    480 ;
0129 344D           481 FUNCT3: CALL    TEST3
012B 0482           482                 JMP             ECHO
                    483 ;
012D 3443           484 FUNCT2: CALL    TEST2
012F 0482           485                 JMP             ECHO
                    486 ;
0131 3435           487 FUNCT1: CALL    TEST1
0133 0482           488                 JMP     ECHD
                    489 ;
                    490 ;*****************************************************************
                    491 ;
                    492 ;TEST1  TENTO KODOVY SEGMENT VYPLNI DISPLAY REGISTRY V RAM OEKTEMENTEM 8
0135 BF08           493 TEST1:  MOV             NEXTPL,#CHARNO
0137 B808           494                 MDV             PNTR0,#CHARND   ;NASTAVENI 8 CYKLOVE SMYCKY
0139 FF             495 TST11:  MOV             A,NEXTPL
013A 14C5           496                 CALL    ENCACC
013C 14DB           497                 CALL    WDISP
013E E839           498                 DJNZ    PNTR0,TST11             ;KOPIROVANI HODNOTY DO DISPLAY REG.
0140 BF08           499                 MDV             NEXTPL,#CHARNO
0142 83             500                 RET
                    501 ;
                    502 ;*****************************************************************
                    503 ;TEST2  ZAPISUJE SEGMENTOVOU VZORKU "AHOJ" DO DISPLAY REG.
                    504 ;               CHVILI POCKA A POTOM VYMAZE OISPLAY
0143 B8C0           505 TEST2:  MOV             PNTR0,#AHOJ
0145 1486           506                 CALL    PRINT
0147 2364           507                 MOV             A,#100                  ;CYKLUS OISPLAY 100 KRAT
0149 3407           508                 CALL    DELAY
014B 04A9           509                 JMP             CLEAR
                    510 ;
                    511 ;*****************************************************************
                    512 ;
                    513 ;TEST3  SUBROUTINA VYPLNUJICI OISPLAY POMLCKAMI
                    514 ;               SKACE DO SUBROUTINY 'CLEAR'
                    515 ;               JAKMILE TLACITKO JE UVOLNENO
014D 2340           516 TEST3:  MOV             A,#01000000B XOR SEGPOL ;VZOREK '-'
014F 14AB           517                 CALL    FILL
0151 3400           518                 CALL    HOLO
0153 04A9           519                 JMP     CLEAR
                    520 ;
                    521 ;*****************************************************************
                    522 ;           TOBOLA JIRI  9.9.1988
                    523 ;*****************************************************************
                    524 END
```

```
USER SYMBOLS
ACCSA  0003   AHOJ   00C0   ASAVE  0002   BLANK  0000   CHARND 0008   CHRPOL 00FF   CHRSTB 0062   CLEAR  00A9
CLR1   00AF   CUROIG 0007   DEBNCE 0004   DELAY  0107   OELAY1 010A   OGPATS 00C8   OPAOO  00F3   ECHO   0082
ENCACC 00C5   ENCMSK 000F   FILL   00AB   FKEY   008C   FUNCT1 0131   FUNCT2 0120   FUNCT3 0129   FUNCT4 0125
FUNCTN 011F   HOLD   0100   INIT   0068   INPMSK 00F0   KBDBUF 0022   KBDIN  008E   KEYLOC 0021   LASTKY 0006
LEGNOS 0099   NCOLS  0004   NEGLOG 00FF   NEXTPL 0007   NREPTS 0020   NROWS  0004   NXTLOC 002E   PDIGIT 0010
PINPUT 0009   PNTR0  0000   PNTR1  0001   POSLOG 0000   PRINT  00B6   PRNT1  00BF   PSGMNT 0008   ROSA   0004
RDELAY 0023   RDPAOO 00F1   REFR1  0013   REFRSH 0010   RENTK1 00EA   RENTRY 00E6   ROTCNT 0005   ROTPAT 0004
SCAN   0029   SCAN1  002C   SCAN3  003F   SCAN5  004A   SCAN6  0050   SCAN8  005A   SCAN9  0062   SEGMAP 0037
SEGPDL 0000   TEST1  0135   TEST2  0143   TEST3  0140   TICK   FFF0   TIINT  0007   TIRET  000E   TST11  0139
WDISP  00DB   WOISP1 00E5

ASSEMBLY COMPLETE,   NO ERRORS
```

Výpis 1. *Zdrojový text obslužného programu*
(922–V1)

PROGRAM PRO MINIMALIZACI LOGICKÉ FUNKCE „BAJT“

Zbyněk Calaba, OK1 SZC, *Krouzova 3039 143 00 Praha 4*

Při návrhu kombinačních logických obvodů se konstruktér často setkává s potřebou vytvořit logickou síť určitých vlastností. Požadovaná funkce se zpravidla vyjadřuje pravdivostní tabulkou pro 2^n stavů, kde n je počet vstupních logických proměnných. Pro určitý počet těchto stavů nabývá logická funkce hodnotu „1“ a pro ostatní „0“. Takovou funkci potom můžeme v triviálním případě vytvořit použitím dekodérů těch stavů, pro které funkce nabývá hodnoty „1“ a provést logický součet jejich výstupů.

Minimalizace logické funkce odpovídá na otázku, zda je vždy nutné při dekódování zahrnovat všechny vstupní proměnné, tj. zda by nebylo možné dva nebo více samostatných dekodérů nahradit jedním.

V praxi existují různé metody pro minimalizaci logických funkcí. Jejich pracnost však prudce roste s počtem vstupních proměnných. A člověk je tvor omylný a pohodlný . . .

V některých případech řeší situaci prostě tak, že k vytvoření funkce použije PROM. Přitom řešení s použitím „klasických“ hradel může vyhovět a navíc odpadnou problémy s programováním paměti.

K usnadnění tohoto rozhodování, pro řešení úlohy minimalizace logických funkcí, případně pro demonstrační účely v kroužcích a kursech, byl vytvořen tento program.

Praktické využití na mikropočítači SPECTRUM je omezeno aplikací na logickou funkci nejvýše osmi vstupních proměnných. Toto omezení je dáno typem „stroje“, neboť celá minimalizace je řešena v paměti omezené délky a v sestavě SPECTRA chybí prostředek pro efektivní práci se soubory.

V amatérské praxi je minimalizace i takové logické funkce přínosem. Kdo nevěří, ať to zkusí manuálně!

Hlavní program plní funkci editoru tabulek popisujících danou logickou funkci. Je napsán v jazyce BASIC-ROM SPECTRUM. Podprogram ve strojovém kódu zajistí velmi rychlý výpočet s výstupem minimalizace ve tvaru „součet součinů“ nebo „negace součtu součinů“. Je využit strojový kód Z-80, délka podprogramu je 1800 bajtů. Při práci programu se využívá celá zbývající paměť počítače.

Použití strojového kódu je nezbytné. V původní variantě byl algoritmus ověřován v jazyce BASIC. Nehledě na další omezení trval výpočet až 30 minut a to i při použití BASIC kompilátorů (BLAST, COLT).

1. Implementace programu

Připusťme, že budoucí uživatel potřebuje především funkční program a že jej nezajímá, co se v paměti počítače děje. Potom se stává otištění výpisu v assembleru neúčelné. Uživatel by musel napsat do počítače podstatně více znaků a protože navíc programu nerozumí (proč také), nemůže ani posoudit, zda pomocí překladače vytvořil správný program, a to i v případě, kdy překladač nehlásí žádné chyby. Pro takového uživatele je zapotřebí, aby mohl na základě údajů autora vytvořit rychle a přesně požadovaný blok údajů v paměti a uchovávat ho na kazetě.

K mnou použitému řešení mne inspiroval způsob používaný v časopise FUNKAMATEUR. Princip spočívá v tom, že je publikován výpis, jehož každý řádek je doplněn kontrolním číslem (je vypočítáno pro každý řádek). Při vkládání údajů do paměti je kontrolní číslo opět vypočítáváno a kontrolováno se zadaným. Uživatel je ihned upozorněn na případnou chybu.

Domnívám se, že tento způsob otiskování programů by si zasloužil rozšíření i u nás. Ušetříme čas a také tiskové strany, kterých se nedostává. Navíc je takový způsob přístupný i „basicovským“ programátorům.

Ostatně, zdatní programátoři si zajisté najdou způsob, jak pořídit zpětný překlad do assembleru, pokud to ovšem potřebují. Nemluvě o programátorské etice, analýza a následná úprava cizího programu je časově náročná.

Ve **Výpisu 4** je uveden program „HEXLOAD“, který umožní vytvoření bloku paměti na základě výpisu výše uvedených vlastností. Tento program je bezpečně použitelný pro vytváření bloků od adresy 6500 H výše.

Program „HEXDUMP“ **(výpis 5)** je určen pro kontrolní výpis paměti na displeji. V případě, že by se navrhovaný způsob prezentace programů ve strojovém kódu ujal, jeho úprava pro tiskárnu nebude činit potíže ani začátečníkům.

Oba programy si vytvářejí v řádku 1 REM strojový kód pro výpočet kontrolního čísla. Využívá se instrukce XOR.

Tyto programy doporučuji vytvořit předem a po ověření funkce uložit na kazetu ještě před zahájením dalších prací.

V paměti je nutno vytvořit dva bloky strojového kódu. Druhý blok je řídicí tabulka. K vytvoření doporučuji následující postup:

1. RESET systému (dojde k vynulování paměti).
2. Zadat postupně: CLEAR 32767, LOAD "HEXLOAD", RUN.
3. Zadat adresu (8000) a uložit první blok (postačí do adresy 8370). Postupně se vždy zadává celý řádek, včetně kontrolního čísla podle šablony v dolní části obrazovky. Pro opravy můžete využívat běžné editační klávesy. V případě správně vložených dat je adresa automaticky zvětšena a ve vkládání můžete pokračovat. Nesprávně vložená data program signalizuje a řádek musíte napsat znova. Po dosažení konce bloku přerušíte program zadáním prázdného vstupního řetězce.
4. Zadáte RUN a celý postup budete opakovat pro druhý blok od adresy 8500.
5. Pořídíte bezpečnostní kopii podprogramu (příkazem SAVE "probajt" CODE 32768, 1800) a provedete její verifikaci.
6. Zadáte NEW (smaže se oblast od 32767 dolů).
7. Vytvoříte hlavní program „BAJT“ podle **výpisu 1.** Pozor! Program se doporučuje spouštět teprve v okamžiku, když je celý vytvořen! V žádném případě to nemůže být dříve, dokud není přítomen v paměti podprogram!
8. Příkazem GOTO 160 pořídíte potom úplnou kopii programu na kazetu.
9. Nyní je možno zahájit testování programu a opravy chyb vzniklých při zápisu programu v BASICu.
10. Konečnou verzi získáte použitím „autosave“ na řádku 160 (viz bod 8).

Poznámka:
Matematikové mohou dokázat, že způsob vytváření kontrolní číslice nezabrání všem možným omylům při vkládání podprogramu do paměti. V případě, že se program při vyvolání minimalizace zhroutí nebo se neočekávaně zachová, mohu doporučit jediné: program „HEXDUMP“ a kontrolu každého bajtu zvlášť.

2. Používání programu

Po zavedení programu je vypsáno základní menu:

1 – tabulku vytvořit,
2 – tabulku číst z magnetofonu,
3 – tabulku uložit na magnetofon,
4 – tabulku opravit,
5 – tabulku zobrazit,
6 – tabulku vytisknout,
7 – tabulku minimalizovat,
8 – tabulku invertovat.

Řádky 3 až 8 jsou zobrazovány pouze tehdy, jestliže program již nějakou tabulku zná.

Program pracuje se soubory (typ 3 – CODE) délky 256 bajtů, jejichž jméno je tvořeno šesti volitelnými znaky a sufixem ".TAB".

Použití tlačítka BREAK přeruší program, s výjimkou případu, kdy pracuje podprogram minimalizace. Pro návrat k menu bez ztráty dat použijte povel GOTO 22. Příkaz RUN také spustí program, avšak způsobí vynulování tabulky.

Hodnoty jsou do programu zadávány v hexadecimálním tvaru. Zadávané hodnoty (řetězce) jsou vždy interpretovány na úrovni posledních dvou pravých znaků, zbytek řetězce je ignorován. Vstup prázdného řetězce je ekvivalentní vstupu „0“ nebo je využit pro ukončení funkce (program vždy upozorní).

Vytvoření tabulky

Tabulka v paměti je vynulována a v následujícím dialogu jsou zadávány termy, tj. stavy, pro které nabývá výstupní logická funkce

hodnotu „1". Celý proces vkládání ukončíte vstupem řetězce nulové délky (stiskem samotné klávesy ENTER). Poté je automaticky zobrazena tabulka, kterou jste vytvořili a jste dotázáni, zda souhlasíte se zadáním (s jejím obsahem).

Pokud odpovíte „A", přejde program zpět k hlavnímu menu. Odpověď „N" vyvolá dotaz, zda si přejete tabulku jen opravit („O") nebo vytvořit znovu („Z"). Rozdíl spočívá v tom, že na pokyn „O" přejde program k funkci 4, pokyn „Z" způsobí vynulování tabulky (návrat k počátku funkce 1). Jiné odpovědi program odmítá.

Uvedený postup je vhodný pro vytvoření tabulky, která je málo obsazena hodnotou „1". Tabulku s převládajícím počtem jedniček získáte snáze tímto postupem:

- bezprostředně po volbě 1 ukončíte práci (ENTER),
- odsouhlasíte nulovou tabulku („A"),
- po zobrazení menu provedete volbu 8,
- použijete volbu 4 – opravit tabulku.

Čtení tabulky z magnetofonu

Po této volbě jste dotázáni na název tabulky, kterou chcete číst. Délka názvu je zkontrolována. Sufix ".TAB" nezadávejte, je doplňován automaticky. Potom zbývá běžná obsluha počítače při čtení souboru. Po úspěšném čtení se program vrací k hlavnímu menu.

V případě, že zadáte prázdný řetězec místo názvu, je nahrán prvý soubor typu 3, ovšem jen za předpokladu, že má shodné parametry jako soubory vytvářené programem „BAJT". Jinak nastane chyba a je nutno program znovu spustit.

Zápis tabulky na magnetofon

Jste požádáni o název tabulky (opět nezadávejte sufix ".TAB"). Vstup prázdného řetězce program odmítá, stejně jako název delší než šest znaků. Další postup je shodný s postupem pro práci se soubory. Po nahrání máte možnost volbou „A" verifikovat uložená data. Jiná volba, stejně jako úspěšná verifikace dat, vede k návratu k hlavnímu menu.

Oprava tabulky

Tabulka je opravována po jednotlivých prvcích, jejichž pozici je vždy nutno zadat. Zadáváte tedy nejdříve pozici, potom je znázorněna stará hodnota a potom jste požádáni o vstup hodnoty nové.

Vstup prázdného řetězce při dotazu na pozici způsobí zobrazení tabulky a žádost o její odsouhlasení. Pozor! Odpověď „Z" vede nenávratně ke ztrátě dat a funkci 1, tj. znovuvytvoření tabulky! Pokud si přejete pokračovat v opravách, je nutno zadat „N" a na následující otázku odpovědět znakem „O".

Zobrazení tabulky na dispieji

Tato volba způsobí, že tabulka je zobrazena ve tvaru mapy, kdy pořadí indexů v řádcích a sloupcích je změněno proti přirozené posloupnosti hexadecimálních číslic. Tento způsob poskytuje konkrétnější grafickou představu o povaze logické funkce a možnostech její minimalizace.

Vytištění tabulky na tiskárně

Výstup je zajištěn příkazem LPRINT. Máte možnost zadat pro tabulku libovolný nadpis. Vzor výpisu je uveden v kontrolním příkladu na **obr. 1.**

Ti uživatelé, kteří nemají k dispozici originální tiskárnu, musí nahrát před spuštěním programu do paměti obslužný podprogram a provést inicializaci. Doporučuji jeho umístění v rozmezí adres 5B00–5BFF, rozhodně však do adresy 7FFF.

```
LOGICKA FUNKCE (OR-AND):

{
 (/A/B/E)
+(/A/B/C/D)
+( A B C D)
}


Tabulka k predchazejici logicke funkci

   0 1 3 2 6 7 5 4 C D F E A B 9 8
   ===============================
0: 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
1: 1 1 1 1 1 1 1 1 0 0 0 0 0 0 0 0
3: 1 1 1 1 1 1 1 1 0 0 0 0 0 0 0 0
2: 1 1 1 1 1 1 1 1 0 0 0 0 0 0 0 0
6: 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
7: 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
5: 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
4: 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
C: 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
D: 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
F: 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
E: 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
A: 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
B: 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
9: 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
8: 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0


LOGICKA FUNKCE (NOR-AND):

/{
 (/A/B/E)
+(/A/B/C/D)
+( A B C D)
 }
```

Obr. 1. Výpisy kontrolního příkladu (921–1)

Minimalizace tabulky

Volbou je spuštěn podprogram minimalizace tabulky logické funkce. Po dobu řešení je zakázáno přerušení. Doba nutná k řešení závisí na obsahu tabulky. Akusticky jste upozorněni, jestliže program nalezl řešení. Po stisknutí libovolné klávesy je řešení vypisováno. Tvar výpisu bude rozebrán dále. Akustická signalizace je pozůstatkem z dřívější doby, je možné ji vypustit. Po ukončení výpisu máte možnost požádat o jeho zopakování (vhodné u delších výpisů).

Invertování tabulky

Volba způsobí záměnu všech prvků „0" tabulky za „1" a obráceně.

3. Kontrolní příklad

Pro ověření základních funkcí vám poslouží následující kontrolní případ. Máme vytvořit kombinační obvod, pro který platí tabulka uvedená uprostřed **obr. 1.** Po spuštění programu zvolte „1" a potom zadávejte:

```
00 01 02 03 04 05 06 07 08 09 0A 0B 0C 0D 0E 0F
10 11 12 13 14 15 16 17
20 21 22 23 24 25 26 27
30 31 32 33 34 35 36 37
F0 F1 F2 F3 F4 F5 F6 F7 F8 F9 FA FB FC FD FE FF
„prázdný vstup".
```

Připomínám, že na pořadí zadávaných hodnot nezáleží, zde prezentovaný tvar vám zadání pouze usnadní. Pokud jste se zmýlili, máte možnost již popsaným postupem tabulku opravit. Na konci celého Vašeho snažení by měla být tabulka zmíněná v úvodu.

Volbou „3" uložte tabulku na pásku. Ověřte i verifikaci dat.

Smazáním tabulky (volba „1" a ihned ENTER) obdržíte nulovou tabulku. Pomocí volby „2" vyzkoušíte čtení z magnetofonu. Musíte obdržet původně vytvořenou tabulku.

Volbou „4" vyzkoušejte opravu tabulky (pokud jste tak již neučinili v kroku vytváření). Zadáte např. AA 1 AA 0. Tabulka je opět původní.

Volbou „8" ověřte inverzi tabulky. Aby bylo dosaženo stejných výsledků jako na obrázku, proveďte inverzi ještě jednou.

Pokud máte připojenou tiskárnu, můžete ověřit tisk tabulky volbou „6". Její zobrazení (volba „5") bylo ověřeno již v mnoha předchozích krocích.

Volba „7" způsobí, že po jisté době bude na váš pokyn zobrazeno řešení ve tvaru uvedeném v horní části obrázku 1.

Nakonec postupnou volbou „8" a „7" obdržíte výsledek v případě invertované tabulky **(obr. 1** dole).

Poznámky k tvaru výpisu logické funkce:

Jednotlivé vstupní proměnné jsou vyjádřeny znaky „A" až „H". Znak „/" před proměnnou vyznačuje negaci proměnné například „/A" znamená „ne A", „A non".

Jednotlivé dílčí součiny jsou uzavřeny v kulatých závorkách, složené závorky uzavírají celou funkci.

Jak již bylo uvedeno, program se snaží šetřit čas a proto v závislosti na rozsahu zadání volí dvě vyjádření logické funkce. Druhý typ výpisu se bude někomu zdát neobvyklý. Využitím pravidel logiky je možno výsledek upravit na aplikaci konkrétně použitých logických prvků. Já jsem si nechtěl komplikovat situaci rozšířením programu pro znázorňování různých variant, za hlavní přínos považuji především zpracování podprogramu minimalizace.

Pokud bude vstupních proměnných méně než osm, zadejte tabulku v menším rozsahu termů a ve výpisu nepřihlížejte k nevyužitým proměnným.

Závěrem přeji všem uživatelům mnoho spokojenosti při práci s tímto programem.

Výpis 1. *Hlavní program* (921–V1)

```
   1 REM ------------------------

   2 REM *BAJT* minimalizace QMC

   3 REM ------------------------

   4 LET pgm=32768: POKE 23658,8

   5 DEF FN a(i)=PEEK (pgm+i)+25
6*PEEK (pgm+i+1)
   6 LET main=FN a(0)
   7 LET t=FN a(2)
   8 RESTORE 14: DIM e(16)
   9 DIM e$(16): DIM q$(8)
  10 FOR i=1 TO 16
  11 READ e(i),e$(i): NEXT i
  12 FOR i=1 TO 8: READ q$(i)
  13 NEXT i
  14 DATA 0,"0",1,"1",3,"3",2,"2
",6,"6",7,"7",5,"5",4,"4"
  15 DATA 12,"C",13,"D",15,"F",1
4,"E",10,"A",11,"B"
  16 DATA 9,"9",8,"8","A","B","C
","D","E","F","G","H"
  17 LET CTRL=2: DIM Z$(8,13)
  18 FOR I=1 TO 8: READ Z$(I)
  19 NEXT I
  20 DATA "VYTVORIT:","CIST Z MG
F","ULOZIT NA MGF","OPRAVIT","ZO
BRAZIT","VYTISKNOUT"
  21 DATA "MINIMALIZOVAT","INVER
TOVAT"
  22 CLS : PRINT ' BRIGHT 1;" **
********** BAJT ********** "
  23 PRINT AT 2,3;"PROGRAM PRACE
S TABULKOU"
  24 PRINT  BRIGHT 1; "   Zbynek
CALABA Praha 4 - 1988 "
  25 PLOT 0,168: DRAW 255,0
  26 DRAW 0,-25: DRAW -255,0
  27 DRAW 0,25
  28 PRINT '''TAB 1; BRIGHT 1;"M
OZNOSTI PROGRAMU:"' BRIGHT 0;" "
```

```
  29 FOR I=1 TO CTRL
  30 PRINT " "; I;" - TABULKU ";
Z$(I): NEXT I
  31 NEXT I
  32 PLOT 0,120: DRAW 255,0
  33 DRAW 0,-90: DRAW -255,0
  34 DRAW 0,90
  35 PRINT #0; FLASH 1; BRIGHT 1
;"VOLTE, prosim!"
  36 BEEP .1,12
  37 LET i$=INKEY$
  38 IF i$="" THEN  GO TO 37
  39 LET i=CODE i$
  40 IF i>56 OR i<48 THEN  GO TO
 37
  41 IF i$="1" THEN  GO SUB 51:
GO TO 22
  42 IF i$="2" THEN  GO TO 106
  43 IF VAL i$ >ctrl THEN  GO TO
 37
  44 IF i$="3" THEN  GO TO 126
  45 IF i$="4" THEN  GO SUB 73:
GO TO 22
  46 IF i$="5" THEN  GO TO 89
  47 IF i$="6" THEN  GO SUB 137:
 GO TO 22
  48 IF i$="7" THEN  GO TO 171
  49 IF i$="8" THEN  GO TO 208
  50 GO TO 37
  51 LET x=1: GO SUB 214
  52 RANDOMIZE USR 33557: REM nu
lovani
  53 PRINT '"Pro popis tabulky z
adavejte term"'"(pozici '1' v ta
bulce)."
  54 PRINT '"Samotny ENTER ukonc
i vstup dat."
  55 BEEP .05,12
  56 LET j$="Prosim term"
  57 GO SUB 113
  58 IF i>255 THEN  GO TO 63
  59 LET m=PEEK (t+i)
  60 IF m = 48 THEN  POKE (t+i),
49: GO TO 55
  61 PRINT #0;"DUPLICITA!"
  62 BEEP .5,1: GO TO 55
  63 GO SUB 90
  64 BEEP .1,5: BEEP .1,9
  65 BEEP .1,12
  66 INPUT "V poradku (A/N)?",i$

  67 IF I$="A" THEN  LET ctrl=8:
 GO TO 22
  68 IF I$<>"N" THEN  BEEP .5,1:
 GO TO 64
  69 BEEP .1,12
  70 INPUT "Opravit - znova (O/Z
)?",i$
  71 IF I$="Z" THEN  GO TO 51
  72 IF I$<>"O" THEN  BEEP .5,1:
 GO TO 69
  73 LET X=4: GO SUB 214
  74 PRINT '"Samotny ENTER ukonc
i vstup dat."
  75 PRINT AT 19,0;"
                    "
  76 BEEP .05,12
  77 LET j$="Pozice v tabulce"
  78 GO SUB 113: LET j=i
  79 IF i>255 THEN  GO TO 63
  80 LET m=PEEK (t+j)
  81 PRINT AT 19,0;"TAB(";i$;")=
";CHR$ m
  82 BEEP .05,12
  83 LET j$="Nova hodnota"
  84 GO SUB 113
  85 IF i=999 THEN, LET i=0
  86 POKE (t+j),48+i
  87 IF i<0 OR i>1 THEN  BEEP .5
,1: GO TO 83
  88 GO TO 75
  89 GO SUB 90: GO TO 123
  90 LET X=5: GO SUB 214
  91 PRINT '" ";: FOR I=1 TO 16
  92 PRINT  BRIGHT 1;E$(I);
  93 NEXT I
  94 PRINT : FOR I=0 TO 15
  95 PRINT  BRIGHT 1;E$(I+1);
  96 FOR J=0 TO 15
  97 LET M=PEEK (t+e(J+1)+16*e(I
+1))
  98 PRINT CHR$ m;
  99 NEXT j: PRINT : NEXT I
 100 RETURN
 101 PRINT #0;"Stop MGF, prosim!
": PAUSE 255: RETURN
 102 INPUT "Jmeno tabulky (max.
6 znaku): ";i$
 103 IF LEN i$>6 THEN  BEEP .5,1
: GO TO 102
 104 IF LEN i$=0 THEN  RETURN
 105 LET i$=i$+".TAB": RETURN
 106 LET X=2: GO SUB 214
 107 GO SUB 102
 108 PRINT AT 3,0;"Jmeno: ";i$
 109 PRINT #0;"Prosim magnetofon
!"
 110 LOAD i$CODE t,256
 111 GO SUB 101
 112 LET ctrl=8: GO TO 22
 113 INPUT (j$+" (hex):"),i$
 114 IF LEN i$=0 THEN  LET i=999
: RETURN
 115 LET i$="0"+i$
 116 IF LEN i$>2 THEN  LET i$=i$
(2 TO ): GO TO 116
 117 LET k=CODE i$(1): LET l=COD
E i$(2)
 118 LET e=k-(48+(k>57)*39)
 119 LET k=e+((e<0)*32)
 120 LET e=l-(48+(l>57)*39)
 121 LET l=e+((e<0)*32)
 122 LET i=16*k+l: RETURN
 123 BEEP .1,12
 124 INPUT "Prosim ENTER!";i$
 125 GO TO 22
 126 LET X=3: GO SUB 214
 127 GO SUB 102
 128 IF LEN i$<>0 THEN  GO TO 13
1
 129 PRINT #0;"Tabulka musi mit
jmeno!"
 130 BEEP .5,1: GO TO 127
 131 PRINT '' "Jmeno: ";i$
 132 SAVE i$CODE t,256
 133 INPUT "Verifikovat (A)?",j$

 134 IF j$<>"A" THEN  GO TO 22
 135 PRINT "Verifikace dat ...":
VERIFY i$CODE t,256
 136 PRINT "Data o.k.": GO TO 12
3
 137 LET X=6: GO SUB 214
 138 GO SUB 153
 139 LPRINT '"   ";
 140 FOR I=1 TO 16:
 141 LPRINT E$(I);" ";
 142 NEXT I: LPRINT
 143 LPRINT "   ";
 144 FOR I=0 TO 15: LPRINT "==";

 145 NEXT I: LPRINT
 146 FOR I=0 TO 15
 147 LPRINT E$(I+1);": ";
 148 FOR J=0 TO 15
 149 LET M=PEEK (t+e(J+1)+16*e(I
+1))
 150 LPRINT m-48;" ";
 151 NEXT j: LPRINT : NEXT I
 152 RETURN
 153 INPUT "Nadpis sestavy:",j$
 154 LPRINT j$: LPRINT
 155 RETURN
 156 CLEAR 32767
 157 PRINT "AUTOSAVE-GO TO 160"
 158 LOAD "probajt"CODE 32768,18
00: RUN
 159 REM autosave programu
 160 CLEAR : SAVE "BAJT" LINE 15
6
 161 SAVE "probajt"CODE 32768,18
00
 162 STOP : GO TO 1
 163 PRINT '''JDE O TRIVIALNI RE
SENI"
 164 BEEP .05,5: BEEP .5,9

 165 BEEP .05,5
 166 PAUSE 255: GO TO 22
 167 BEEP 1,1
 168 PRINT '' "Zjistena chyba pr
ogramu."
 169 PRINT "Doporucuji nahrat pr
ogram znovu."
 170 PRINT "a pred reklamaci opa
kovat.": PAUSE 0: GO TO 22
 171 LET X=7: GO SUB 214
 172 PRINT ''"Cekejte, prosim!"'
''"Doba reseni zavisi na slozito
sti"
 173 PRINT "tabulky. Reseni ohla
sim!"
 174 RANDOMIZE USR main
 175 LET od=FN a(4)
 176 LET do=FN a(6)
 177 LET rc=PEEK (pgm+8)
 178 IF rc=1 THEN  GO TO 163
 179 IF rc=3 THEN  GO TO 181
 180 IF rc<>0 THEN  GO TO 167
 181 CLS : PRINT '' FLASH 1; BRI
GHT 1;"R E S E N I !"
 182 PRINT  BRIGHT 1''"Stisknete
 klavesu."
 183 BEEP .05,5: BEEP .05,9
 184 IF INKEY$="" THEN  GO TO 18
3
 185 IF rc=3 THEN  GO TO 201
 186 CLS : PRINT  BRIGHT 1;"LOGI
CKA FUNKCE (OR-AND):": PRINT
 187 PRINT '"{": GO SUB 191: PRI
NT "}"
 188 GO SUB 211
 189 IF i$="A" THEN  GO TO 186
 190 GO TO 22
 191 LET i=od: PRINT " (";
 192 LET j=PEEK (i+1)
 193 FOR k=1 TO 8: LET j=PEEK (i
+k)
 194 IF j=45 THEN  GO TO 197
 195 IF j=48 THEN  PRINT "/";q$(
k);: GO TO 197
 196 IF j=49 THEN  PRINT " ";q$(
k);
 197 NEXT k: PRINT ")"
 198 LET i=i+10
 199 IF i<>do THEN  PRINT "+(";:
 GO TO 193
 200 RETURN
 201 CLS : PRINT  BRIGHT 1;"LOGI
CKA FUNKCE (NOR-AND):"
 202 PRINT : PRINT  BRIGHT 1;"/{
"
 203 GO SUB 191
 204 PRINT  BRIGHT 1;" }": BEEP
.05,12
 205 GO SUB 211
 206 IF i$="A" THEN  GO TO 201
 207 RANDOMIZE USR 33572: GO TO
22
 208 LET X=8: GO SUB 214
 209 RANDOMIZE USR 33572: PAUSE
10'
 210 GO TO 89
 211 INPUT "Opakovat vypis resen
i (A)? ";i$
 212 RETURN
 213 BEEP .1,12: CLS : RETURN
 214 GO SUB 213: PRINT  BRIGHT 1
;"TABULKU ";z$(x)
 215 RETURN
```

Výpis 2. Podprogram (921–V2)

```
Vypis s kontrolou
8000 09 80 00 84 3B 88 4F 88:79
8008 03 F3 FD 22 00 88 21 00:86
8010 84 7E 2C 28 05 BE 20 07:62
8018 18 F8 3E 01 C3 5E 82 AF:6F
8020 32 08 80 21 00 84 01 00:1E
8028 00 3E 31 BE 20 01 04 23:B7
8030 0D 20 F8 3E 80 B8 30 1C:FF
8038 3E 03 32 08 80 21 00 84:22
8040 01 31 30 7E B8 20 03 71:94
8048 18 05 B9 C2 57 82 70 23:E0
8050 2C 2D 20 EF 21 00 00 22:CD
8058 37 88 22 39 88 3E 01 B8:AB
8060 28 B8 AF 32 00 87 01 FF:74
8068 00 11 01 87 21 00 87 ED:DC
8070 B0 21 3B 88 22 04 80 22:A6
8078 06 80 11 00 84 1A FE 31:C6
8080 20 09 21 00 86 6B 6E 26:AD
8088 87 36 FF 13 1C 1D 20 ED:91
8090 11 00 87 2A 06 80 1A 3C:1C
8098 20 26 D5 16 85 1A 01 08:53
80A0 00 E5 23 07 F5 38 04 3E:36
80A8 30 18 03 3E 31 04 77 23:74
```

```
80B0 F1 0D 20 EF AF 77 E1 70:7A
80B8 01 0A 00 09 22 06 80 D1:77
80C0 13 1C 1D 20 D1 CD C6 82:6A
80C8 FE 00 C2 57 82 2A 06 80:45
80D0 22 08 88 22 06 88 3E 08:38
80D8 F5 FB 3E 7F DB FE 1F 38:4D
80E0 03 C3 56 82 F3 ED 4B 06:47
80E8 80 2A 04 80 CD 67 82 CA:CC
80F0 56 82 3E 08 11 13 88 ED:85
80F8 53 02 88 F5 2A 02 88 7E:F2
8100 23 B6 CA FD 81 23 7E 23:5D
8108 B6 CA FD 81 DD 2A 02 88:7D
8110 DD 6E 00 DD 66 01 E5 DD:31
8118 2A 02 88 DD 5E 02 DD 56:AA
8120 03 D5 E5 D5 E5 D5 AF 08:71
8128 AF 32 0A 88 11 0B 88 D5:58
8130 23 DD E1 01 08 00 ED B0:4B
8138 0E 08 D1 E1 E5 D5 EB DD:30
8140 2B 13 23 DD 23 1A BE 28:69
8148 15 08 3C FE 02 28 21 08:DC
8150 FE 2D 28 18 3E 2D BE 28:66
8158 05 DD 77 00 18 0E DD 7E:1A
8160 00 FE 31 20 07 3A 0A 88:50
8168 3C 32 0A 88 0D 20 D2 08:7B
8170 DD E1 FD E1 FE 01 20 53:AC
8178 3E FF DD 77 09 FD 77 09:E1
8180 E5 D5 C5 ED 4B 08 88 2A:F9
8188 06 88 CD 67 82 28 23 22:8F
8190 04 88 0E 09 11 0A 88 1A:02
8198 BE 20 07 13 23 0D 20 F7:73
81A0 18 26 01 0A 00 2A 04 88:93
81A8 09 ED 4B 08 88 CD 67 82:07
81B0 20 DD 2A 08 88 E5 01 0A:B9
81B8 00 09 22 08 88 D1 01 09:72
81C0 00 21 0A 88 ED B0 AF 12:43
81C8 C1 D1 E1 D1 E1 D1 EB DD:26
81D0 2A 02 88 DD 46 15 DD 4E:BD
81D8 14 CD 67 82 28 08 01 0A:17
81E0 00 09 EB C3 21 81 E1 DD:BD
81E8 2A 02 88 DD 46 13 DD 4E:BB
81F0 12 CD 67 82 28 07 01 0A:1E
81F8 00 09 C3 16 81 CD 8F 82:9D
8200 F1 2A 02 88 23 23 22 02:71
8208 88 3D C2 FB 80 CD 8F 82:CC
8210 2A 06 80 ED 5B 04 80 37:A9
8218 3F ED 52 E5 C1 2A 08 88:0E
8220 ED 5B 06 88 37 3F ED 52:8F
8228 CD 67 82 20 18 2A 04 80:BE
8230 ED 5B 06 88 ED 4B 06 80:18
8238 1A BE 20 09 23 13 CD 67:17
8240 82 28 18 18 F3 CD 6D 82:7B
8248 CD C6 82 FE 00 20 07 F1:A1
8250 3D C2 D8 80 18 01 E1 3E:61
8258 02 18 03 F1 18 03 32 08:C9
8260 80 FD 2A 00 88 FB C9 7C:91
8268 B8 C0 7D B9 C9 ED 4B 06:D5
8270 88 C5 ED 5B 04 80 2A 08:5D
8278 88 37 3F ED 42 E5 C1 E1:EA
8280 ED B0 ED 53 06 88 ED 53:D3
8288 06 80 ED 53 08 88 C9 DD:AC
8290 2A 02 88 DD 6E 00 DD 66:A8
8298 01 7D B4 C8 E5 FD E1 3E:C7
82A0 FF FD BE 09 28 10 ED 5B:3B
82A8 08 88 01 0A 00 ED B0 ED:3B
82B0 53 08 88 FD E5 E1 DD 46:B1
82B8 13 DD 4E 12 CD 67 82 C8:72
82C0 11 0A 00 19 18 D6 FD E5:D4
82C8 DD 21 11 88 FD 21 23 88:12
82D0 21 13 88 AF 77 E5 D1 13:45
82D8 01 24 00 ED B0 2A 04 80:D6
82E0 11 0A 00 ED 4B 06 80 3E:05
82E8 FF 08 08 BE 28 16 DD 23:81
82F0 DD 23 FD 23 FD 23 3C FE:3C
82F8 09 28 17 BE 20 F0 DD 75:F0
8300 00 DD 74 01 FD 75 00 FD:DD
8308 74 01 08 19 CD 67 82 20:6C
8310 D9 AF FD E1 C9 3E 30 21:8C
8318 00 84 E5 D1 13 77 01 FF:2A
8320 00 ED B0 C9 21 00 84 01:30
8328 31 30 7E B8 20 03 71 18:8D
8330 01 70 23 2C 2D 20 F3 C9:49
8338 00 00 00 00 00 00 00 00:00
8340 00 00 00 00 00 00 00 00:00
8348 00 00 00 00 00 00 00 00:00
8350 00 00 00 00 00 00 00 00:00
8358 00 00 00 00 00 00 00 00:00
8360 00 CD 8C 82 00 00 2A 06:EF
8368 80 C3 10 82 00 00 00 00:D1
8370 00 00 00 00 00 00 00 00:00
8378 00 00 00 00 00 00 00 00:00
8380 00 00 00 00 00 00 00 00:00
8388 00 00 00 00 00 00 00 00:00
8390 00 00 00 00 00 00 00 00:00
8398 00 00 00 00 00 00 00 00:00
83A0 00 00 00 00 00 00 00 00:00
83A8 00 00 00 00 00 00 00 00:00
83B0 00 00 00 00 00 00 00 00:00
83B8 00 00 00 00 00 00 00 00:00
83C0 00 00 00 00 00 00 00 00:00
83C8 00 00 00 00 00 00 00 00:00
83D0 00 00 00 00 00 00 00 00:00
83D8 00 00 00 00 00 00 00 00:00
83E0 00 00 00 00 00 00 00 00:00
83E8 00 00 00 00 00 00 00 00:00
83F0 00 00 00 00 00 00 00 00:00
83F8 00 00 00 00 00 00 00 00:00
```

Výpis 3. Tabulka (921–V3)

```
Vypis s kontrolou
8500 00 01 02 04 08 10 20 40:7F
8508 80 03 05 06 09 0A 0C 11:9E
8510 12 14 18 21 22 24 28 30:21
8518 41 42 44 48 50 60 81 82:3C
8520 84 88 90 A0 C0 07 0B 0D:FD
8528 0E 13 15 16 19 1A 1C 23:22
8530 25 26 29 2A 2C 31 32 34:1B
8538 38 43 45 46 49 4A 4C 51:66
8540 52 54 58 61 62 64 68 70:21
8548 83 85 86 89 8A 8C 91 92:0C
8550 94 98 A1 A2 A4 A8 B0 C1:72
8558 C2 C4 C8 D0 E0 0F 17 1B:FD
8560 1D 1E 27 2B 2D 2E 33 35:0A
8568 36 39 3A 3C 47 4B 4D 4E:06
8570 53 55 56 59 5A 5C 63 65:09
8578 66 69 6A 6C 71 72 74 78:06
8580 87 8B 8D 8E 93 95 96 99:06
8588 9A 9C A3 A5 A6 A9 AA AC:09
8590 B1 B2 B4 B8 C3 C5 C6 C9:06
8598 CA CC D1 D2 D4 D8 E1 E2:0A
85A0 E4 E8 F0 1F 2F 37 3B 3D:FD
85A8 3E 4F 57 5B 5D 5E 67 6B:72
85B0 6D 6E 73 75 76 79 7A 7C:0C
85B8 8F 97 9B 9D 9E A7 AB AD:21
85C0 AE B3 B5 B6 B9 BA BC C7:66
85C8 CB CD CE D3 D5 D6 D9 DA:1B
85D0 DC E3 E5 E6 E9 EA EC F1:22
85D8 F2 F4 F8 3F 5F 6F 77 7B:FD
85E0 7D 7E 9F AF B7 BB BD BE:3C
85E8 CF D7 DB DD DE E7 EB ED:21
85F0 EE F3 F5 F6 F9 FA FC 7F:9E
85F8 BF DF EF F7 FB FD FE FF:7F
8600 00 01 02 09 03 0A 0B 25:2D
8608 04 0C 0D 26 0E 27 28 5D:7F
8610 05 0F 10 29 11 2A 2B 5E:7D
8618 12 2C 2D 5F 2E 60 61 A3:C0
8620 06 13 14 2F 15 30 31 62:58
8628 16 32 33 63 34 64 65 A4:E5
8630 17 35 36 66 37 67 68 A5:EF
8638 38 69 6A A6 6B A7 A8 DB:22
8640 07 18 19 39 1A 3A 3B 6C:48
8648 1B 3C 3D 6D 3E 6E 6F A9:E1
8650 1C 3F 40 70 41 71 72 AA:FB
8658 42 73 74 AB 75 AC AD DC:46
8660 1D 43 44 76 45 77 78 AE:88
8668 46 79 7A AF 7B B0 B1 DD:4D
8670 47 7C 7D B2 7E B3 B4 DE:53
8678 7F B5 B6 DF B7 E0 E1 F7:E2
8680 08 1E 1F 48 20 49 4A 80:E2
8688 21 4B 4C 81 4D 82 83 B8:53
8690 22 4E 4F 84 50 85 86 B9:4D
8698 51 87 88 BA 89 BB BC E2:88
86A0 23 52 53 8A 54 8B 8C BD:46
86A8 55 8D 8E BE 8F BF C0 E3:FB
86B0 56 90 91 C1 92 C2 C3 E4:E1
86B8 93 C4 C5 E5 C6 E6 E7 F8:48
86C0 24 57 58 94 59 95 96 C7:22
86C8 5A 97 98 C8 99 C9 CA E8:EF
86D0 5B 9A 9B CB 9C CC CD E9:E5
86D8 9D CE CF EA D0 EB EC F9:58
86E0 5C 9E 9F D1 A0 D2 D3 ED:C0
86E8 A1 D4 D5 EE D6 EF F0 FA:7D
86F0 A2 D7 D8 F1 D9 F2 F3 FB:7F
86F8 DA F4 F5 FC F6 FD FE FF:2D
```

Výpis 4. Program „HEXLOAD" (921–V4)

```
  1 REM ****************konec
  2 FOR i=23760 TO 23776
  3 READ a: POKE i,a: NEXT i
  4 DATA 175,50,224,92,201,58
  5 DATA 224,92,79,58,223,92
  6 DATA 169,24,242,0,0
  7 POKE 23658,8
  8 POKE 23609,120
  9 DIM B(8)
 10  INPUT "ADRESA (HEX):";Ax
 11 IF LEN Ax<4 THEN  LET Ax="0
"+Ax: GO TO 11
 12 IF LEN Ax>4 THEN  LET Ax=Ax
(2 TO ): GO TO 12
 13 LET I=1: GO SUB 48
 14 IF D<0 OR D>255 THEN  GO TO
 10
 15 LET A=256*D
 16 LET I=3: GO SUB 48
 17 IF D<0 OR D>255 THEN  GO TO
 10
 18 LET A=A+D
 19 CLS : PRINT "ADRESA: ";Ax;"
 = ";A
 20 LET A=A-1
 21  INPUT " ------DATA------ -
-"'Ax: LET K=0
 22 IF LEN Ax=0 THEN  STOP
 23 IF LEN Ax<>19 THEN  GO TO 4
5
 24 IF Ax(17)<>" " THEN  GO TO
45
 25 FOR I=1 TO LEN Ax-4 STEP 2
 26 GO SUB 48
 27 IF D<0 OR D>255 THEN  GO TO
 45
 28 LET K=K+1: LET B(K)=D
 29 NEXT I
 30  REM KONTROLNI SUMA
 31 LET I=18: GO SUB 48
 32 IF D<0 OR D>255 THEN  GO TO
 45
 33 REM TEST
 34 RANDOMIZE USR 23760
 35 FOR I=1 TO 8
 36 POKE 23775,B(I)
 37 RANDOMIZE USR 23765
 38 NEXT I: LET I=PEEK 23776
 39 IF I<>D THEN  GO TO 45
 40 REM ULOZIT
 41 FOR I=1 TO 8
 42 POKE (A+I),B(I)
 43 NEXT I: LET A=A+9
 44 GO SUB 51: GO TO 20
 45 PRINT Ax'"... CHYBA!"
 46 BEEP 1,1: GO TO 21
 47  REM CTENI HEX
 48 LET D=((CODE Ax(I)-(48+(COD
E Ax(I)>57)*7)))*16
 49 LET D=D+((CODE Ax(I+1)-(48+
(CODE Ax(I+1)>57)*7)))
 50  RETURN
 51 CLS : PRINT "ADRESA: ";
 52 LET D=INT (A/256)
 53 GO SUB 57
 54 LET D=A-256*D
 55 GO SUB 57
 56 PRINT " = ";A: RETURN
 57 LET Ax=""
 58 LET A1=INT (D/16)
 59 LET A2=D-16*A1
 60  LET Ax=CHR$ (48+A1+((A1>9)
*7))
 61  LET Ax=Ax+CHR$ (48+A2+((A2
>9)*7))
 62 PRINT Ax;: RETURN
```

Výpis 5. Program „HEXDUMP" (921–V5)

```
  1 REM ****************konec
  2 POKE 23658,8
  3 POKE 23609,120
  4 PRINT "Vypis s kontrolou"
  5 FOR I=23760 TO 23776
  6 READ A: POKE I,A: NEXT I
  7 DATA 175,50,224,92,201,58
  8 DATA 224,92,79,58,223,92
  9 DATA 169,24,242,0,0
 10 POKE 23658,8: POKE 23609,1
 11 INPUT "START(HEX):";Ax
 12 GO SUB 40
 13 IF A<0 THEN  BEEP 1,1: GO T
O 11
```

```
  14 LET Q1=A
  15 INPUT "STOP (HEX):";Ax
  16 GO SUB 40
  17 IF A<0 THEN  BEEP 1,1: GO T
O 15
  18 LET Q2=A
  19 FOR I=Q1 TO Q2 STEP 8
  20 RANDOMIZE USR 23760
  21 LET A=INT (I/256)
  22 GO SUB 33
  23 LET A=I-256*A: GO SUB 33
  24 FOR J=0 TO 7
  25 LET A=PEEK (I+J)
  26 POKE 23775,A
  27 RANDOMIZE USR 23765
  28 PRINT " ";: GO SUB 33
  29 NEXT J: PRINT ":";
  30 LET A=PEEK 23776
  31 GO SUB 33: PRINT
  32 NEXT I: STOP
  33 LET Ax=""
  34 LET A1=INT (A/16)
  35 LET A2=A-16*A1
  36 LET Ax=CHRx (48+A1+((A1>9)*
7))
  37 LET Ax=Ax+CHRx (48+A2+((A2>
9)*7))
  38 PRINT Ax;
  39 RETURN
  40 IF LEN Ax<4 THEN  LET Ax="0
"+Ax: GO TO 40
  41 IF LEN Ax>4 THEN  LET Ax=Ax
(2 TO ): GO TO 41
  42 LET I=1: GO SUB 48
  43 GO SUB 47
  44 LET A=256*D: LET I=3
  45 GO SUB 48: LET A=A+D
  46 GO SUB 47: RETURN
  47 IF D<0 OR D>255  THEN  LET
A=-1: RETURN
  48  LET D=((CODE ax(i)-(48+(CO
DE ax(i)>57)*7)))*16
  49  LET D=D+((CODE ax(i+1)-(48
+(CODE ax(i+1)>57)*7)))
  50 RETURN
```

MINIDATA

Martin Preisler, *Rudé armády 684, 293 01 Mladá Boleslav*

Při vytváření databází jsem začal přemýšlet jak do paměti SPECTRA vložit více dat, aniž bych rozšiřoval paměť. Nejdříve jsem zkrátil program, ale moc místa jsem tím neušetřil. Pak jsem si uvědomil, že délka jednoho ASCII znaku je 8 bitů, přičemž všechny nejsou využity (stačí si prohlédnout tabulku ASCII).

Proto jsem se rozhodl vytvořit nový znakový soubor – ve zkrácené formě. Nazval jsem jej MDAT (minidata). Abych ušetřil co největší prostor paměti, zvolil jsem jako délku jednoho znaku 5 bitů. To umožňuje ale pouze 32 znaků. Rozdělil jsem tedy údaje v databankách na nenumerické (skládají se z písmen A–Z, čárky, tečky, mezery, dvojtečky, pomlčky, apostrofu) a číselné. To vyžaduje dvojí sadu znaků.

Třicet dva číslic neexistuje, proto jsem číselné znaky doplnil matematickými operátory (krát, plus, mínus, lomeno, šipka, levá a pravá závorka, desetinná tečka), některými funkcemi vkládanými v rozšířeném módu (COS, INT, ABS, SIN, SQR, LN, EXP, PI, ASN, ACS) a písmeny (A, B, C, E). Což umožní uchovávat nejrůznější vzorce pro proměnné A, B, C (E označuje exponent) a při použití funkce VAL získat přímo hodnoty.

Pak jsem musel vymyslet program, který by normální ASCII znaky převáděl do paměti jako MDAT a naopak. Aby byl program rychlý, vytvořil jsem ho ve strojovém kódu mikroprocesoru Z80. Vycházel jsem z **obr. 1.** Takto uložených 8 znaků by se stále opakovalo. Proto jsem rozdělil program na 16 částí (8 pro zápis do paměti a 8 pro čtení) podle příznaku. Program jsem se snažil maximálně zjednodušit. Znaky, které nejsou v dané tabulce znaků, program přepíše na „–" (mínus). Významy některých symbolů:

ZAPIS –adresa začátku programu pro zápis dat;

PRECTI –adresa začátku rutiny pro čtení dat;

TAB –adresa obsahuje adresu používané tabulky nově vytvořených znaků. V mém případě PISMO (nenumerické znaky) = adresa TAB+357 a CISLA (numerické) = adresa TAB+421. Tabulku si můžete vytvořit svou vlastní (i více), ale zachovávejte pořadí – znak, kód, posledním znakem musí být mínus;

SLOVO –adresa obsahuje adresu, z které se při zápisu bude číst ASCII řetězec a na kterou se při čtení bude zapisovat ASCII řetězec;

DELKA –adresa obsahuje délku řetězce, z toho vyplývá omezení pro délku na 255 znaků;

ADR –adresa obsahuje adresu, na kterou se bude zapisovat, nebo ze které se bude číst;

PRIZ –adresa obsahuje příznak, který určuje přesné místo zápisu, resp. čtení, které bude použito (viz **obr. 1).**

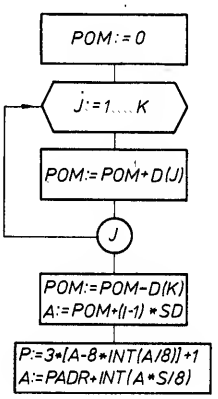
Při stálém zapisování resp. čtení si program obsahy ADR a PRIZ mění sám. Když použijete konstantní obsah SLOVO, stačí vkládat délku a použitou tabulku. Při zapisování, resp. čtení z prostředka (třídění, vyhledávání, přerušené vkládání) můžete využít algoritmus pro výpočet adresy a příznaku. Obecně vypočítáme K-tou položku I-té věty (viz **obr. 2).**

Hlavní výhody programu:

- maximální využití paměti,
- šetří 38% paměti,
- pro zpracování je přístupno větší množství dat.

Nevýhody:

- pomalejší proces čtení, zápisu a vyhledávání (pomalejší neznamená pomalý, časové ztráty jsou vcelku zanedbatelné),
- nemožnost použití číslic a písmen v jednom údaji.

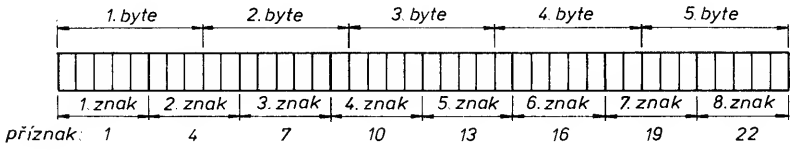


Obr. 2. Algoritmus pro výpočet adresy a příznaku (919–2)
(D(J) – délka J-tého údaje, PADR – první adresa datového souboru, P – příznak, A – adresa pro zápis-čtení znaku).

***Výpis 1.** Zdrojový text programu* (919-V1)

```
919-V1
                TAB     64000
                SLOVO   TAB+2
                ADR     TAB+4
                DELKA   TAB+6
                PRIZ    TAB+7
                ORG     TAB+8

ZAPIS   CALL    ZAC
A1      PUSH    BC
        LD      C,(HL)
        PUSH    HL
        LD      HL,(TAB)
        CALL    NAJDI
        INC     HL
        LD      C,(HL)
        LD      HL,(ADR)
        LD      A,(HL)
```



Obr. 1. Princip ukládání znaků (919–1)

```
        LD      IX,A2+1
        ADD     IX,DE
A2      JP      (IX)
        JP      Z1
        JP      Z2
        JP      Z3
        JP      Z4
        JP      Z5
        JP      Z6
        JP      Z7
        JP      Z8
A3      LD      (ADR),HL
A4      LD      (HL),A
        INC     E
        INC     E
        INC     E
A5      POP     HL
        INC     HL
        POP     BC
        DJNZ    A1
        LD      HL,PRIZ
        LD      (HL),E
        RET
Z1      AND     11100000B
        ADD     A,C
        JP      A4
Z2      AND     00011111B
        LD      B,A
        LD      A,C
        LD      C,00H
        RLCA
        RLA
        RLA
        RLA
        RL      C
        RLCA
        RL      C
        AND     11100000B
        ADD     A,B
        LD      (HL),A
        INC     HL
        LD      A,(HL)
        AND     11111100B
        ADD     A,C
        JP      A3
Z3      AND     10000011B
        RLC     C
        RLC     C
        ADD     A,C
        JP      A4
Z4      RLCA
        RR      C
        RRA
        LD      (HL),A
        INC     HL
        LD      A,(HL)
        AND     11110000B
        ADD     A,C
        JP      A3
Z5      LD      B,00H
        AND     00001111B
        RLC     C
        RLC     C
        RLC     C
        SLA     C
        RL      B
        ADD     A,C
        LD      (HL),A
        INC     HL
        LD      A,(HL)
        AND     11111110B
        ADD     A,B
        JP      A3
Z6      AND     11000001B
        RLC     C
        ADD     A,C
        JP      A4
Z7      AND     00111111B
        LD      B,A
        LD      A,00H
        RR      C
        RRA
        RRC     C
        RRA
        ADD     A,B
        LD      (HL),A
        LD      A,C
        AND     00000111B
        LD      B,A
        INC     HL
        LD      A,(HL)
        AND     11111000B
        ADD     A,B
        JP      A3
Z8      AND     00000111B
        RLC     C
        RLC     C
        RLC     C
        ADD     A,C
        LD      (HL),A
        INC     HL
        LD      (ADR),HL
        LD      E,01H
        JP      A5

PRECTI  CALL    ZAC
B1      PUSH    BC
        PUSH    HL
        LD      HL,(ADR)
        LD      A,(HL)
        LD      IX,XX+1
        ADD     IX,DE
XX      JP      (IX)
        JP      B3
        JP      C2
        JP      C3
        JP      C4
        JP      C5
        JP      C6
        JP      C7
        JP      C8
B2      LD      (ADR),HL
B3      INC     E
        INC     E
        INC     E
B4      AND     00011111B
        LD      C,A
        LD      HL,(TAB)
        INC     HL
        CALL    NAJDI
        DEC     HL
        LD      A,(HL)
        POP     HL
        LD      (HL),A
        INC     HL
        POP     BC
        DJNZ    B1
        LD      HL,PRIZ
        LD      (HL),E
        RET

ZAC     LD      D,00H
        LD      A,(DELKA)
        LD      B,A
        LD      HL,PRIZ
        LD      E,(HL)
        LD      HL,(SLOVO)
        RET

NAJDI   LD      A,(HL)
        CP      45
        RET     Z
        CP      C
        RET     Z
        INC     HL
        INC     HL
        JP      NAJDI
C2      INC     HL
        LD      B,(HL)
        RR      B
        RRA
C52     RR      B
        RRA
        RRA
        RRA
        RRA
        JP      B2
C3      RRA
C6      RRA
        JP      B3
C4      RLCA
        INC     HL
        LD      A,(HL)
        RLA
        JP      B2
C5      INC     HL
        LD      B,(HL)
        JP      C52
C7      INC     HL
        LD      B,(HL)
        RLCA
        RL      B
        RLCA
        RL      B
        LD      A,B
        JP      B2
C8      RRA
        RRA
        RRA
        INC     HL
        LD      (ADR),HL
        LD      E,1
        JP      B4

PISMO   DEFB    'A',0
        DEFB    'B',1
        DEFB    'C',2
        DEFB    'D',3
        DEFB    'E',4
        DEFB    'F',5
        DEFB    'G',6
        DEFB    'H',7
        DEFB    'I',8
        DEFB    'J',9
        DEFB    'K',10
        DEFB    'L',11
        DEFB    'M',12
        DEFB    'N',13
        DEFB    'O',14
        DEFB    'P',15
        DEFB    'Q',16
        DEFB    'R',17
        DEFB    'S',18
        DEFB    'T',19
        DEFB    'U',20
        DEFB    'V',21
        DEFB    'W',22
        DEFB    'X',23
        DEFB    'Y',24
        DEFB    'Z',25
        DEFB    ' ',26
        DEFB    ':',27
        DEFB    '.',28
        DEFB    ',',29
        DEFB    '"',30
        DEFB    '-',31

CISLO   DEFB    0,0
        DEFB    1,1
        DEFB    2,2
        DEFB    3,3
        DEFB    4,4
        DEFB    5,5
        DEFB    6,6
        DEFB    7,7
        DEFB    8,8
        DEFB    9,9
        DEFB    '*',10
        DEFB    '.',11
        DEFB    '+',12
        DEFB    '^',13
        DEFB    '/',14
        DEFB    '(',15
        DEFB    ')',16
        DEFB    'A',17
        DEFB    'B',18
        DEFB    'C',19
        DEFB    'E',20
        DEFB    179,21;COS
        DEFB    186,22;INT
        DEFB    189,23;ABS
        DEFB    178,24;SIN
        DEFB    187,25;SQR
        DEFB    184,26;LN
        DEFB    185,27;EXP
        DEFB    167,28;PI
        DEFB    181,29;ASN
        DEFB    182,30;ACS
        DEFB    '-',31
```

GRAFIKA V PASCALU

Pavel Kříž, *Paláskova 1107/2, 180 00 Praha 8*

Grafika je souhrn podprogramů tvořících grafický systém. S jejich pomocí můžete snadno vytvářet různé obrázky, schémata, grafy apod. Podprogramy umožňují transformovat souřadný systém, kreslit čáry a jednoduché obrazce, měnit barvy, vyplňovat uzavřené obrazce. Kreslit můžete na celou plochu obrazovky o velikosti 256×192 bodů.

Systém je vytvořen jako knihovna podprogramů přizpůsobená pro Hisoft Pascal 4T. Tuto knihovnu můžete dále podle své potřeby rozšiřovat.

Přehled podprogramů

Podle činnosti lze podprogramy rozdělit do pěti částí:

1) Organizační podprogramy:

GINIT	nastaví implicitní hodnoty,
WINDOW	definuje velikost okénka pro kreslení,
CLS	vymaže obrazovku,
CLW	vymaže okénko,
LTYPE	definuje druh čar,
DTYPE	definuje způsob zobrazení bodů a čar,
PATTERN	definuje vzorek pro vyplňování a jeho rotaci,
AT	nastaví polohu tisku,
INK,PAPER	
BRIGHT,FLASH	nastaví barvy pro kreslení,
BORDER	změní barvu okraje obrazovky.

2) Transformační podprogramy:

SCALE	nastaví měřítka os,
COORD	nastaví počátek souřadného systému,
TURN	natočí souřadný systém,
SYMMET	nastaví symetrii,
SHIFT	provede posunutí od počátku COORD.

3) Kreslicí podprogramy:

CURSOR	nastaví kurzor,
PLOT	nakreslí bod,
DRAW,LINETO,	
LINE	nakreslí přímku,
TRIA	nakreslí trojúhelník,
TETRA	nakreslí čtyřúhelník,
BOX	nakreslí obdélník rovnoběžný s osami x a y,
BOW	nakreslí oblouk,
CIRCLE	nakreslí kružnici,
FILL	vyplní uzavřený obrazec,
PAINT	vyplní uzavřený obrazec zadaným vzorkem.

4) Funkce:

POSIT	posice kurzoru,
POINT	informace o bodu.

5) Interně používané procedury a funkce:

V této části jsou podprogramy přizpůsobení a pomocné podprogramy používané v částech 1 až 4. Pro úplnost uvádím i jejich seznam:

CURSOR0, DRAW0, FILL0, INVERSE, OVER, PAINT0, POINT0, RANGE, ROT, TRANS, TRCONST.

Systém používá globální proměnné:

GDTYPE, GLTYPE, GPAT, GROT, XMAN, XMIN, XOS, XSYM, YMAX, YMIN, YOS, YSYM: INTEGER;
GALFA, GDETER, XCO, XSC, XSH, XTR1, XTR2, XTR3, YCO, YSC, YSH, YTR1, YTR2, YTR3: REAL.

Popis podprogramů

1) Organizační podprogramy:

GINIT

Tento podprogram musíte vyvolat před použitím ostatních podprogramů. GINIT provede nastavení implicitních hodnot:

WINDOW (0,0,255,191,0,0);	celá obrazovka.
SCALE (1,1);	měřítko 1:1 pro x i y.
COORD (0,0,);	počátek zůstane
SHIFT (0,0);	v levém dolním rohu.
SYMMET (0,0); TYRN (0);	osy jsou totožné s okraji obrazovky.
DTYPE (1); LTYPE (1);	plná čára.
PATTERN (1,1);	jednoduchý vzorek.

WINDOW (XMIN,YMIN,XMAX,YMAX, X0,Y0: INTEGER)

Definuje polohu okénka na obrazovce. Střed souřadnic se nastaví ve vzdálenosti |X0,Y0| od levého dolního rohu okénka. Parametry jsou nezávislé na nastaveném měřítku.

Parametry musíte zadat tak, aby platilo

0 <= XMIN <= XMAX <= 255 0 <= X0 <= XMAX-XMIN
0 <= YMIN <= YMAX <= 255 0 <= X0 <= YMAN-YMIN

CLS

Vymaže obrazovku, přenese dočasné barvy do trvalých a vyplní jimi obrazovku. Nastaví tisk do levého horního rohu, kurzor pro kreslení nastaví do bodu |0,0|.

CLW (FRAME: BOOLEAN)

Vymaže okénko definované příkazem WINDOW. Parametrem FRAME si zvolíte, má-li se okénko ohraničit rámečkem (TRUE).

LTYPE (LT: INTEGER)

Definuje druh čar:
1 ..... plná čára,
2 ..... čárkovaná čára,
3 ..... tečkovaná čára.

DTYPE (DT: INTEGER)

Definuje způsob zobrazení bodů a čar:
0 ..... vymazávání (INVERSE 1),
1 ..... plná čára,
2 ..... změna bodů (OVER 1).

PATTERN (PAT,RP: INTEGER)

Těmito parametry se řídí způsob vyplňování uzavřeného obrazce podprogramem PAINT.

PAT je osmibitový vzorek, který je v binárním tvaru „naskládán" do vodorovné linky.

RP ovlivňuje rotaci vzoru při přechodu na vyšší linku. Vzorek se posune o RP bitů vpravo (pro RP<0 o |RP| bitů vlevo).

Parametry musí být v rozmezí
0 <= PAT <= 255
–7 <= RP <= 7.
Příklady vzorků: BIN 01010101 = 85,
BIN 00010001 = 17.

AT (LIN,COL: INTEGER)

Nastaví tiskovou pozici na řádek LIN a sloupec COL.

INK (I:INTEGER)
PAPER (P:INTEGER)
BRIGHT (B:INTEGER)
FLASH (F:INTEGER)

Těmito podprogramy se nastaví dočasně aktivní barvy. Dovolené parametry jsou 0 až 9 pro INK a PAPER, 0,1,8 pro BRIGHT a FLASH. Význam parametrů viz manuál ZX Spectra.

BORDER (B: INTEGER)

Tímto podprogramem změníte barvu okraje obrazovky. Dovolené hodnoty jsou 0 až 7.

2) Transformační podprogramy:

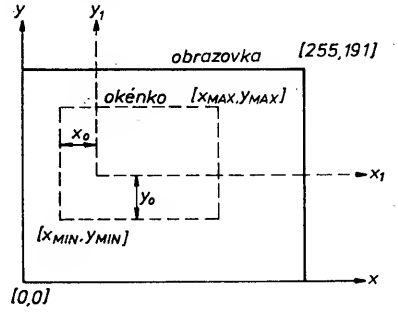
SCALE (SX,SY: REAL)

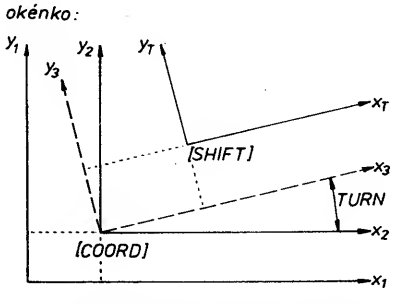
Nastaví měřítko SX:1 pro osu x a SY:1 pro osu Y. Nepřípustným parametrem je 0. Měřítko ovlivňuje dále uvedené transformace.

COORD (X0,Y0: REAL)

Nastaví počátek souřadného systému. X0,Y0 je posunutí od středu zadaného podprogramem WINDOW.

K bodu COORD se vztahují dále uvedené transformace.





Obr. 1. Znázornění transformací (926–1)

TURN (ALFA: REAL)
Souřadný systém se otočí okolo bodu COORD o ALFA stupňů (proti směru hodinových ručiček).

SYMMET (SX,SY: INTEGER)
Nastaví se symetrie:
(1,0) ..... podle osy x,
(0,1) ..... podle osy y,
(1,1) ..... podle počátku,
(0,0) ..... zrušení symetrie.

SHIFT (SX,SY: REAL)
V natočeném svstému souřadnic se bod |0,0| posune ve vzdálenosti |SX,SY| od bodu COORD.

Uvedené transformační podprogramy provádějí pouze nastavování potřebných hodnot. Při opakovaném volání platí vždy poslední zadaná hodnota, na níž předchozí nemá žádný vliv. Na pořadí volání transformačních podprogramů nezáleží.

Transformace se zruší vyvoláním podprogramu s nulovými hodnotami (pro měřítko s hodnotami (1,1)).

Znázornění transformací je na **obr. 1.**

3) Kreslicí podprogramy:

CURSOR (X,Y: REAL)
Přenese kurzor do bodu |X,Y|.

V dále uvedených podprogramech se kurzor nastaví do posledního nakresleného bodu, u obrazců (TRIA, TETRA, BOX) zůstane v prvním zadaném bodě, po vyplnění (FILL, PAINT) se nastaví do zadaného vnitřního bodu.

PLOT (X,Y: REAL)
Nakreslí bod |X,Y|.

DRAW (DX,DY: REAL)
Nakreslí přímku od kurzoru do bodu |Xkurzoru+DX, Ykurzoru+DY|.

LINETO (X2,Y2: REAL)
Nakreslí přímku od kurzoru do bodu |X2, Y2|.

LINE (X1,Y1,X2,Y2: REAL)
Nakreslí přímku z bodu |X1,Y1| do bodu |X2,Y2|.

TRIA (X1,Y1,X2,Y2,X3,Y3,X4,Y4: REAL)
Nakreslí trojúhelník zadaný třemi body.

TETRA (X1,Y1,X2,Y2,X3,Y3,X4,Y4: REAL)
Nakreslí čtyřúhelník zadaný čtyřmi body.

BOX (X1mY1,X2,Y2: REAL)
Nakreslí obdélník rovnoběžný s osami x a y, přičemž body |X1,Y1|, |X2,Y2| jsou souřadnice úhlopříčky.

BOW (X,Y,R,ALFA,BETA: REAL)
Nakresli oblouk se středem v bodě |X,Y| a poloměrem R. Začátek oblouku svírá s poloosou X+ úhel ALFA. BETA je vnitřní úhel oblouku. (Hodnoty ALFA, BETA jsou ve stupních.)

CIRCLE (X,Y,R: REAL)
Nakreslí kružnici se středem v |X,Y| a poloměrem R. Pokud budou měřítka pro osu x a y rozdílná, bude výsledkem ve skutečnosti elipsa.

FILL (X,Y: REAL)
Vyplní uzavřený obrazec barvou inkoustu. Bod |X,Y| je uvnitř tohoto obrazce.

PAINT (X,Y: REAL)
Vyplní uzavřený obrazec vzorkem zadaným podprogramem PATTERN. Bod |X,Y| leží opět uvnitř tohoto obrazce.

Poznámka: Při větších a složitějších obrazcích jsou podprogramy FILL a PAINT náročnější na volnou paměť. Tyto požadavky ovlivníte kladně tím, že vnitřní bod zadáte uprostřed vyplňovaného obrazce.

Parametry všech kreslicích podprogramů se zadávají v transformovaných souřadnicích. Pokud bude bod ležet mimo zadané okénko, vypíše se zpráva Out of window X= . . . Y= . . . XT= . . . YT= . . ., kde XT a YT jsou hodnoty transformovaných souřadnic, X a Y jsou hodnoty přepočítané do skutečných souřadnic. Po vypsání této chybové zprávy se program zastaví se zprávou Halt at PC= . . .

4) Funkce:

POSIT (VAR,PX,PY: REAL)
Ve skutečnosti nejde o funkci, ale o proceduru, která vrací pozici kurzoru v transformovaných souřadnicích.

POINT (X,Y: REAL): BOOLEAN
Tato funkce vrací informaci o bodu:
TRUE .... bod je vyplněn barvou inkoustu,
FALSE ... bod není vyplněn.

Použití grafiky

Tato verze systému grafiky je přizpůsobena pro Hisoft Pascal. Zdrojový text systému s názvem grafR.P je dlouhý 7485 bajtů. Vytvořený cílový kód je dlouhý 8715 bajtů.

Po nahrání Hisoft Pascalu musíte zadat tabulku symbolů (Table size). Celý grafický systém spotřebuje pro tabulku 2600 bajtů. Bohužel většina verzí HP dovoluje použít pro tabulku symbolů max. 2200 bajtů. Potřebnou hodnotu 2600 i více lze zadat u verze HP 113 (P. Adámek).

Uvedené hodnoty naznačují, že v paměti nezbývá již mnoho místa (HP zabírá přes 20 kB). Zdrojový text grafiky proto nelze kompilovat celý. Z programu můžete vymazat části, které nebudete potřebovat a pak zkompilovat.

Aby se paměť nemusela zbytečně zatěžovat uchováváním zdrojového textu, je výhodné použít kompilaci z pásky. To umožňuje kompilovat celý grafický systém. Proto jsem vytvořil „knihovní verzi".

Systém grafika je zde rozdělen do 5 částí:

1 **grafR0.W** – velikost cílového kódu je 3506 bajtů.
Základní část, obsahující podprogramy přizpůsobení, organizační podprogramy (GINIT, CLS, LTYPE, DTYPE, PATTERN, AT), kreslící podprogramy (CURSOR, PLOT, DRAW, LINE, LINETO);

2. **grafR1.W** – 811 bajtů.
Obsahuje organizační podprogramy (WINDOW, CLW), transformační podprogramy (SCALE, COORD, TURN, SYMMET, SHIFT);

3. **grafR2.W** – 2031 bajtů.
Obsahuje kreslicí podprogramy (TRIA, TETRA, BOX, BOW, CIRCLE);

4. **grafR3.W** – 2217 bajtů.
Obsahuje kreslicí podprogramy (FILLm PAINT);

5. **colour.W** – 150 bajtů.
Obsahuje podprogramy barev.

Poznámka: označení .W v názvu znamená, že nejde o standardní formát zdrojového textu, ale o knihovní verzi nahranou editorovým povelem W.

Při použití GRAFIKY zkompilujte část grafR0. a k ní pak přidejte další potřebnou část. Mezi deklaraci proměnných a deklaraci procedur a funkcí ve svém programu vložte řádky:

{$SF grafR0.W}
{$SF grafRx.W},

kde x je číslo části.

Pokud nemáte ve svém programu žádnou proměnnou, dejte před tento řádek klíčové slovo VAR.

Při spuštění kompilace editorovým povelem C nebo T se u řádku s direktivou F kompilace zastaví a je připravena na vkládání z pásky.

Verze Integer

Pro jednodušší kreslení jsem vytvořil verzi INTEGER, která je kratší a vytváří i menší cílový kód. Zdrojový kód s názvem grafI.P je dlouhý 5744 bajtů, velikost cílového kódu je 5376 bajtů. Pro tabulku symbolů stačí 2200 bajtů. Vyhovuje tedy všem verzím Hisoft Pascalu.

Odlišnosti od verze REAL:

I. Kreslit můžete na obrazovku o velikosti 256×176 bodů.

II. Všechny uvedené podprogramy, které měly parametry typu REAL, mají zde parametry typu INTEGER.

III. Verze Integer neobsahuje tyto podprogramy:
LTYPE – definice typu čar,
SCALE – nastavení měřítek,
TURN – natočení souřadného systému,
BOW – kreslení oblouku.

IV. Knihovna má následující části:

1. **grafI0.W** – velikost cílového kódu je 1894 bajtů.
Základní část, obsahující podprogramy přizpůsobení, organizační podprogramy (GINIT, CLS, DTYPE, PATTERN AT), kreslicí podprogramy (CURSOR, PLOT, DRAW, LINE, LINETO, CIRCLE);

2. **grafI1.W** – 584 bajtů.
Obsahuje organizační podprogramy (WINDOW, CLW), transformační podprogramy (COORD, SYMMET, SHIFT);

3. **grafI2.W** – 604 bajtů.
Obsahuje kreslicí podprogramy (TRIA, TETRA, BOX);

4. **grafI3.W** – 2144 bajtů.
Obsahuje kreslicí podprogramy (FILL, PAINT);

5. **colour.W** – 150 bajtů.
Obsahuje podprogramy barev (stejně jako u verze REAL).

Všechny programy, napsané pro verzi Integer, budou pracovat stejně i s verzí Real.

Ukázky programů

1. **Schody (Výpis 3.)**
Tento program ukazuje použití okénka, změnu měřítka, transformace souřadnic a vyplňování.

2. **Průběh funkce (Výpis 4.)**
Nakreslí se průběh jedné periody funkce Y=SIN (X). Ukazuje výhody změny měřítka.

3. **Pneumatika (Výpis 5.)**
Program nakreslí zjednodušeně pneumatiku a vybarví ji. Je použita verze INTEGER grafického systému.

Výpis 1. *Grafický systém, verze REAL*
(926–VI)

```
{    GRAFICKY SYSTEM    }
{    P.KRIZ (c) 1988    }
{      verze  REAL      }

{$L-}
XMIN,YMIN,XMAX,YMAX,XOS,YOS,XSYM,YSYM,
GPAT,GROT,GDTYPE,GLTYPE :INTEGER;
XCO,YCO,XSC,YSC,XSH,YSH,GALFA,GDETER,
XTR1,XTR2,XTR3,YTR1,YTR2,YTR3 :REAL;
{$O-,C-,S-,I-}
PROCEDURE CURSOR0(X,Y:INTEGER);
BEGIN
 INLINE(221,102,2,221,110,4,34,125,92)
END;
PROCEDURE PLOT0(X,Y:INTEGER);
BEGIN
 INLINE(253,33,58,92,221,70,2,221,78,4,237);
 INLINE(67,125,92,62,191,205,172,34,205,236,34)
END;
PROCEDURE DRAW0(DX,DY:INTEGER);
BEGIN
 INLINE(253,33,58,92,17,1,1,221,70,2);
 INLINE(221,126,3,167,40,4,87,175,144,71);
 INLINE(221,78,4,221,126,5,167,40,4,95);
 INLINE(175,145,79,121,184,48,6,105,213,175);
 INLINE(95,24,8,177,40,51,104,65,213,22);
 INLINE(0,96,120,31,133,56,3,188,56,7);
 INLINE(148,79,217,193,197,24,4,79,213,217);
 INLINE(193,42,125,92,121,133,79,120,132,71);
 INLINE(237,67,125,92,62,191,205,172,34,205);
 INLINE(236,34,217,121,16,214,209)
END;
FUNCTION POINT0(X,Y:INTEGER):BOOLEAN;
BEGIN
 INLINE(221,70,2,221,78,4,62,191,205,172,34);
 INLINE(71,4,126,7,16,253,230,1,221,119,6)
END;
PROCEDURE POSIT0(VAR PX,PY:INTEGER);
BEGIN
 INLINE(237,91,125,92,175,221,110,4);
 INLINE(221,102,5,115,35,119,221,110,2);
 INLINE(221,102,3,114,35,119)
END;
PROCEDURE OVER(O:INTEGER);
BEGIN
 INLINE(62,2,221,86,2,205,29,34)
END;
PROCEDURE INVERSE(I:INTEGER);
BEGIN
 INLINE(62,1,221,86,2,205,29,34)
END;
PROCEDURE AT(LIN,COL:INTEGER);
BEGIN
 INLINE(221,70,2,221,78,4,205,10,32)
END;
PROCEDURE TRANS(X,Y:REAL;VAR PX,PY:INTEGER);
BEGIN
 PX:=ROUND(XTR1+XTR2*X+XTR3*Y);
 PY:=ROUND(YTR1+YTR2*X+YTR3*Y);
 IF (PX<XMIN) OR (PX>XMAX) OR (PY<YMIN) OR (PY>YMAX) THEN
  BEGIN
   WRITE(CHR(13),'Out of window: X=',PX,' Y=',PY,' XT=',X,
YT=',Y);HALT
  END
END;
PROCEDURE POSIT(VAR PX,PY:REAL);
VAR X,Y:INTEGER;
BEGIN
 POSIT0(X,Y);
 PX:=((X-XTR1)*YTR3-(Y-YTR1)*XTR3)/GDETER;
 PY:=((Y-YTR1)*XTR2-(X-XTR1)*YTR2)/GDETER
END;
PROCEDURE TRCONST;
BEGIN
 XTR2:=YSYM*XSC*COS(GALFA);
 YTR2:=YSYM*XSC*SIN(GALFA);
 XTR3:=-XSYM*YSC*SIN(GALFA);
 YTR3:=XSYM*YSC*COS(GALFA);
 XTR1:=XOS+XSC*XCO+XTR2*XSH+XTR3*YSH;
 YTR1:=YOS+YSC*YCO+YTR2*XSH+YTR3*YSH;
 GDETER:=XTR2*YTR3-XTR3*YTR2
END;
PROCEDURE LTYPE(LT:INTEGER);
BEGIN
 GLTYPE:=LT
END;
PROCEDURE DTYPE(DT:INTEGER);
BEGIN
 GDTYPE:=DT;
 OVER(0);INVERSE(0);
 IF DT=0 THEN INVERSE(1);
 IF DT=2 THEN OVER(1)
END;
PROCEDURE PATTERN(PAT,RP:INTEGER);
BEGIN
 GPAT:=PAT;GROT:=RP
END;
FUNCTION POINT(X,Y:REAL):BOOLEAN;
VAR PX,PY:INTEGER;
BEGIN
 TRANS(X,Y,PX,PY);
 POINT:=POINT0(PX,PY)
END;
PROCEDURE CURSOR(X,Y:REAL);
VAR PX,PY:INTEGER;
BEGIN
 TRANS(X,Y,PX,PY);
 CURSOR0(PX,PY)
END;
PROCEDURE PLOT(X,Y:REAL);
VAR PX,PY:INTEGER;
BEGIN
 TRANS(X,Y,PX,PY);
 PLOT0(PX,PY)
END;
PROCEDURE DRAW(DX,DY:REAL);
VAR A,B,N,X,Y,I:INTEGER;
    XR,YR,PDX,PDY,L:REAL;
BEGIN
 POSIT(XR,YR);
 TRANS(XR+DX,YR+DY,X,Y);
 PDX:=XTR2*DX+XTR3*DY;
 PDY:=YTR2*DX+YTR3*DY;
 IF GLTYPE=1 THEN DRAW0(ROUND(PDX),ROUND(PDY))
 ELSE BEGIN
  IF GLTYPE=2 THEN BEGIN A:=2;B:=4 END
  ELSE BEGIN A:=4;B:=0 END;
  L:=SQRT(SQR(PDX)+SQR(PDY));
  N:=TRUNC((L+B)/(A+B));
  POSIT0(X,Y);
  FOR I:=0 TO N-1 DO BEGIN
   PLOT0(X+ROUND(PDX*I/N),Y+ROUND(PDY*I/N));
   IF GLTYPE=2 THEN
    DRAW0(ROUND(PDX*A/L),ROUND(PDY*A/L))
  END;
  PLOT0(X+ROUND(PDX),Y+ROUND(PDY))
 END
END;
PROCEDURE LINETO(X2,Y2:REAL);
VAR X1,Y1:REAL;
BEGIN
 POSIT(X1,Y1);DRAW(X2-X1,Y2-Y1)
END;
PROCEDURE LINE(X1,Y1,X2,Y2:REAL);
BEGIN
 PLOT(X1,Y1);DRAW(X2-X1,Y2-Y1)
END;
PROCEDURE CLS;
BEGIN
 PAGE;AT(0,0);CURSOR(0,0);
 INLINE(205,173,28,58,141,92,33,0,88);
 INLINE(17,1,88,1,255,2,119,237,176)
END;
PROCEDURE GINIT;
BEGIN
 XSC:=1;YSC:=1;XCO:=0;YCO:=0;GALFA:=0;
 XSH:=0;YSH:=0;XSYM:=1;YSYM:=1;XOS:=0;YOS:=0;
 XMIN:=0;YMIN:=0;XMAX:=255;YMAX:=191;
 TRCONST;
 CURSOR(0,0);
 DTYPE(1);LTYPE(1);
 PATTERN(1,1)
END;
PROCEDURE WINDOW(X1,Y1,X2,Y2,X0,Y0:INTEGER);
BEGIN
 XMIN:=X1;YMIN:=Y1;XMAX:=X2;YMAX:=Y2;XOS:=XMIN
+X0;YOS:=YMI N+Y0:
 TRCONST
END;
PROCEDURE COORD(X0,Y0:REAL);
BEGIN
 XCO:=X0;YCO:=Y0;TRCONST
END;
PROCEDURE SCALE(SX,SY:REAL);
BEGIN
 XSC:=SX;YSC:=SY;TRCONST
END;
PROCEDURE SHIFT(SX,SY:REAL);
BEGIN
 XSH:=SX;YSH:=SY;TRCONST
END;
PROCEDURE SYMMET(SX,SY:INTEGER);
BEGIN
 IF SX=0 THEN XSYM:=1 ELSE XSYM:=-1;
 IF SY=0 THEN YSYM:=1 ELSE YSYM:=-1;
 TRCONST
END;
PROCEDURE TURN(ALFA:REAL);
BEGIN
 GALFA:=ALFA*3.1416/180;TRCONST
END;
PROCEDURE CLW(FRAME:BOOLEAN);
VAR Y,T:INTEGER;
BEGIN
 T:=GDTYPE;DTYPE(0);
 FOR Y:=YMIN TO YMAX DO BEGIN
  PLOT0(XMIN,Y);DRAW0(XMAX-XMIN,0)
 END;
 DTYPE(T);
 IF FRAME THEN BEGIN
  CURSOR0(XMIN,YMIN);
  DRAW0(0,YMAX-YMIN);DRAW0(XMAX-XMIN,0);
  DRAW0(0,YMIN-YMAX);DRAW0(XMIN-XMAX,0)
 END;
 CURSOR(0,0)
END;
PROCEDURE TRIA(X1,Y1,X2,Y2,X3,Y3:REAL);
BEGIN
 CURSOR(X1,Y1);
 DRAW(X2-X1,Y2-Y1);
 DRAW(X3-X2,Y3-Y2);
 DRAW(X1-X3,Y1-Y3)
END;
PROCEDURE TETRA(X1,Y1,X2,Y2,X3,Y3,X4,Y4:REAL);
BEGIN
 CURSOR(X1,Y1);
 DRAW(X2-X1,Y2-Y1);
 DRAW(X3-X2,Y3-Y2);
 DRAW(X4-X3,Y4-Y3);
 DRAW(X1-X4,Y1-Y4)
END;
PROCEDURE BOX(X1,Y1,X2,Y2:REAL);
BEGIN
 CURSOR(X1,Y1);
 DRAW(0,Y2-Y1);
 DRAW(X2-X1,0);
 DRAW(0,Y1-Y2);
 DRAW(X1-X2,0)
END;
PROCEDURE BOW(X,Y,R,ALFA,BETA:REAL);
VAR U,V,RX,RY,RM:REAL;
BEGIN
 ALFA:=ALFA*3.1416/180;
 BETA:=BETA*3.1416/180;
 CURSOR(X+R*COS(ALFA),Y+R*SIN(ALFA));
 RX:=ABS(R*XSC);RY:=ABS(R*YSC);
 IF RX>RY THEN RM:=RX ELSE RM:=RY;
 U:=3.1416/36;
 IF RM<10 THEN U:=2*U;
 IF RM<25 THEN U:=3*U;
```

```
240  V:=U;
241  WHILE V<=BETA DO BEGIN
242   LINETO(X+R*COS(ALFA+V),Y+R*SIN(ALFA+V));
243   V:=V+U
244  END;
245  LINETO(X+R*COS(ALFA+BETA),Y+R*SIN(ALFA+BETA));
246 END;
247 PROCEDURE CIRCLE(X,Y,R:REAL);
248 BEGIN
249  BOW(X,Y,R,0,360)
250 END;
251 PROCEDURE RANGE(VAR X1,X2:INTEGER; Y:INTEGER);
252 BEGIN
253  IF X1>XMIN THEN
254   REPEAT X1:=X1-1
255   UNTIL (X1=XMIN) OR POINT0(X1,Y);
256  IF X2<XMAX THEN
257   REPEAT X2:=X2+1
258   UNTIL (X2=XMAX) OR POINT0(X2,Y)
259 END;
260 {$S+}
261 PROCEDURE FILL0(VAR X:INTEGER; Y:INTEGER);
262 VAR X1,X2:INTEGER;
263 BEGIN
264  X2:=X;
265  RANGE(X,X2,Y);
266  CURSOR0(X,Y);
267  X:=X+1;
268  DRAW0(X2-X,0);
269  X1:=X;
270  IF Y<YMAX THEN
271   WHILE X<X2 DO BEGIN
272    IF NOT POINT0(X,Y+1) THEN FILL0(X,Y+1);
273    X:=X+1
274   END;
275  X:=X1;
276  IF Y>YMIN THEN
277   WHILE X<X2 DO BEGIN
278    IF NOT POINT0(X,Y-1) THEN FILL0(X,Y-1);
279    X:=X+1
280   END;
281  X:=X2;
282 END;
283 {$S-}
284 PROCEDURE FILL(X,Y:REAL);
285 VAR PX,PY,X1,T:INTEGER;
286 BEGIN
287  TRANS(X,Y,PX,PY);
288  IF NOT POINT0(PX,PY) THEN BEGIN
289   T:=GDTYPE;DTYPE(1);
290   X1:=PX;
291   FILL0(PX,PY);
292   DTYPE(T)
293  END;
294  CURSOR0(X1,PY)
295 END;
296 {$S+}
297 PROCEDURE PAINT0(X,Y,PAT:INTEGER);
298 VAR X1,X2,PAT1:INTEGER;

299  FUNCTION ROT(PAT,N:INTEGER):INTEGER;
300  VAR I:INTEGER;
301  BEGIN
302   FOR I:=-1 DOWNTO N DO BEGIN
303    PAT:=PAT*2;
304    IF PAT>=255 THEN PAT:=PAT-255
305   END;
306   FOR I:=1 TO N DO BEGIN
307    IF PAT MOD 2=1 THEN PAT:=PAT+256;
308    PAT:=PAT DIV 2
309   END;
310   ROT:=PAT
311  END;
312 BEGIN
313  X1:=X;X2:=X;
314  RANGE(X1,X2,Y);
315  X1:=X1+1;X2:=X2-1;
316  PAT:=ROT(PAT,(X-X1) MOD 8);
317  PAT1:=PAT;
318  FOR X:=X1 TO X2 DO BEGIN
319   PAT:=PAT*2;
320   IF PAT>255 THEN BEGIN
321    PAT:=PAT-255;
322    PLOT0(X,Y)
323   END
324  END;
325  IF Y<YMAX THEN BEGIN
326   PAT:=ROT(PAT1,XSYM*YSYM*GROT);
327   FOR X:=X1 TO X2 DO BEGIN
328    IF (PAT>127) AND NOT POINT0(X,Y+1) THEN PAINT0(X,Y+1,PA
T);
329    PAT:=PAT*2;
330    IF PAT>255 THEN PAT:=PAT-255
331   END
332  END;
333  IF Y>YMIN THEN BEGIN
334   PAT:=ROT(PAT1,-XSYM*YSYM*GROT);
335   FOR X:=X1 TO X2 DO BEGIN
336    IF (PAT>127) AND NOT POINT0(X,Y-1) THEN PAINT0(X,Y-1,PA
T);
337    PAT:=PAT*2;
338    IF PAT>255 THEN PAT:=PAT-255
339   END
340  END
341 END;
342 {$S-}
343 PROCEDURE PAINT(X,Y:REAL);
344 VAR PX,PY,T:INTEGER;
345 BEGIN
346  TRANS(X,Y,PX,PY);
347  IF NOT POINT0(PX,PY) THEN BEGIN
348   T:=GDTYPE;DTYPE(1);
349   PAINT0(PX,PY,GPAT);
350   DTYPE(T)
351  END;
352  CURSOR0(PX,PY)
353 END;
354 PROCEDURE INK(I:INTEGER);
355 BEGIN
356  INLINE(55,221,86,2,205,52,34)
357 END;
358 PROCEDURE PAPER(P:INTEGER);
359 BEGIN
360  INLINE(167,221,86,2,205,52,34)
361 END;
362 PROCEDURE BRIGHT(B:INTEGER);
363 BEGIN
364  INLINE(151,221,86,2,205,116,34)
365 END;
366 PROCEDURE FLASH(F:INTEGER);
367 BEGIN
368  INLINE(175,60,221,86,2,205,29,34)
369 END;
370 PROCEDURE BORDER(B:INTEGER);
371 BEGIN
372  INLINE(221,126,2,205,151,34)
373 END;
374 {$L+,O+,C+,S+,I-}
```

Výpis 2. *Grafický systém, verze INTEGER* (926–V2)

```
1
2 {    GRAFICKY SYSTEM    }
3 {    P.KRIZ (c) 1988    }
4 {     verze INTEGER     }
5
6 {$L-}
7 XCO,YCO,XSYM,YSYM,XMIN,YMIN,XMAX,YMAX,
8 XOS,YOS,XSH,YSH,GPAT,GROT,GDTYPE: INTEGER;
9 {$O-,C-,S-,I-}
10 PROCEDURE CURSOR0(X,Y:INTEGER);
11 BEGIN
12  INLINE(221,102,2,221,110,4,34,125,92)
13 END;
14 PROCEDURE PLOT0(X,Y:INTEGER);
15 BEGIN
16  INLINE(253,33,58,92,221,70,2,221,78,4,205,229,34)
17 END;
18 PROCEDURE DRAW0(DX,DY:INTEGER);
19 BEGIN
20  INLINE(253,33,58,92,17,1,1,221,70,2);
21  INLINE(221,126,3,167,40,4,87,175);
22  INLINE(144,71,221,78,4,221,126,5,167);
23  INLINE(40,4,95,175,145,79,205,188,36)
24 END;
25 PROCEDURE CIRCLE0(X,Y,R:INTEGER);
26 BEGIN
27  INLINE(221,126,6,205,40,45);
28  INLINE(221,126,4,205,40,45);
29  INLINE(221,126,2,205,40,45);
30  INLINE(205,45,35)
31 END;
32 FUNCTION POINT0(X,Y:INTEGER):BOOLEAN;
33 BEGIN
34  INLINE(221,70,2,221,78,4);
35  INLINE(205,170,34,71,4,126,7);
36  INLINE(16,253,230,1,221,119,6)
37 END;
38 PROCEDURE POSIT0(VAR PX,PY:INTEGER);
39 BEGIN
40  INLINE(237,91,125,92,175,221,110,4);
41  INLINE(221,102,5,115,35,119,221,110,2);
42  INLINE(221,102,3,114,35,119)
43 END;
44 PROCEDURE OVER(O:INTEGER);
45 BEGIN
46  INLINE(62,2,221,86,2,205,29,34)
47 END;
48 PROCEDURE INVERSE(I:INTEGER);
49 BEGIN
50  INLINE(62,1,221,86,2,205,29,34)
51 END;
52 PROCEDURE AT(LIN,COL:INTEGER);
53 BEGIN
54  INLINE(221,70,2,221,78,4,205,10,32)
55 END;
56 PROCEDURE TRANS(VAR PX,PY:INTEGER);
57 BEGIN
58  PX:=(PX+XSH)*YSYM+XCO+XOS;
59  PY:=(PY+YSH)*XSYM+YCO+YOS;
60  IF (PX<XMIN) OR (PX>XMAX) OR (PY<YMIN) OR (PY>YMAX) THEN
61   BEGIN WRITE(CHR(13),'Out of window: X=',PX,'Y=',PY);HALT
END
62 END;
63 FUNCTION POINT(X,Y:INTEGER):BOOLEAN;
64 BEGIN
65  TRANS(X,Y);POINT:=POINT0(X,Y)
66 END;
67 PROCEDURE POSIT(VAR PX,PY:INTEGER);
68 BEGIN
69  POSIT0(PX,PY);
70  PX:=(PX-XCO-XOS)*YSYM-XSH;
71  PY:=(PY-YCO-YOS)*XSYM-YSH
72 END;
73 PROCEDURE CURSOR(X,Y:INTEGER);
74 BEGIN
75  TRANS(X,Y);CURSOR0(X,Y)
76 END;
77 PROCEDURE PLOT(X,Y:INTEGER);
78 BEGIN
79  TRANS(X,Y);PLOT0(X,Y)
80 END;
81 PROCEDURE DRAW(DX,DY:INTEGER);
82 VAR X,Y:INTEGER;
83 BEGIN
84  POSIT(X,Y);X:=X+DX;Y:=Y+DY;
85  TRANS(X,Y);
86  DRAW0(DX*YSYM,DY*XSYM)
87 END;
88 PROCEDURE LINETO(X2,Y2:INTEGER);
89 VAR X1,Y1:INTEGER;
90 BEGIN
91  POSIT(X1,Y1);DRAW(X2-X1,Y2-Y1)
92 END;
93 PROCEDURE LINE(X1,Y1,X2,Y2:INTEGER);
94 BEGIN
95  PLOT(X1,Y1);DRAW(X2-X1,Y2-Y1)
96 END;
97 PROCEDURE CIRCLE(X,Y,R:INTEGER);
```

```
  98 BEGIN
  99  TRANS(X,Y);CIRCLE0(X,Y,R)
 100 END;
 101 PROCEDURE DTYPE(T:INTEGER);
 102 BEGIN
 103  GDTYPE:=T;
 104  OVER(0);INVERSE(0);
 105  IF T=0 THEN INVERSE(1);
 106  IF T=2 THEN OVER(1)
 107 END;
 108 PROCEDURE PATTERN(PAT,RP:INTEGER);
 109 BEGIN
 110  GPAT:=PAT;GROT:=RP
 111 END;
 112 PROCEDURE CLS;
 113 BEGIN
 114  PAGE;AT(0,0);CURSOR(0,0);
 115  INLINE(205,173,28,58,141,92,33,0,88);
 116  INLINE(17,1,88,1,255,2,119,237,176)
 117 END;
 118 PROCEDURE GINIT;
 119 BEGIN
 120  XMIN:=0;YMIN:=0;XMAX:=255;YMAX:=175;
 121  XOS:=0;YOS:=0;XCO:=0;YCO:=0;XSH:=0;YSH:=0;
 122  XSYM:=1;YSYM:=1;
 123  CURSOR(0,0);
 124  DTYPE(1);
 125  PATTERN(1,1)
 126 END;
 127 PROCEDURE WINDOW(X1,Y1,X2,Y2,X0,Y0:INTEGER);
 128 BEGIN;
 129  XMIN:=X1;YMIN:=Y1;XMAX:=X2;YMAX:=Y2;XOS:=X1+X0;
YOS:=Y1+Y0
 130 END;
 131 PROCEDURE COORD(X0,Y0:INTEGER);
 132 BEGIN
 133  XCO:=X0;YCO:=Y0
 134 END;
 135 PROCEDURE SYMMET(SX,SY:INTEGER);
 136 BEGIN
 137  IF SX=0 THEN XSYM:=1 ELSE XSYM:=-1;
 138  IF SY=0 THEN YSYM:=1 ELSE YSYM:=-1
 139 END;
 140 PROCEDURE SHIFT(SX,SY:INTEGER);
 141 BEGIN
 142  XSH:=SX;YSH:=SY
 143 END;
 144 PROCEDURE CLW(FRAME:BOOLEAN);
 145 VAR Y,T:INTEGER;
 146 BEGIN
 147  T:=GDTYPE;DTYPE(0);
 148  FOR Y:=YMIN TO YMAX DO BEGIN,
 149   PLOT0(XMIN,Y);DRAW0(XMAX-XMIN,0)
 150  END;
 151  DTYPE(T);
 152  IF FRAME THEN BEGIN
 153   CURSOR0(XMIN,YMIN);
 154   DRAW0(0,YMAX-YMIN);DRAW0(XMAX-XMIN,0);
 155   DRAW0(0,YMIN-YMAX);DRAW0(XMIN-XMAX,0)
 156  END;
 157  CURSOR(0,0)
 158 END;
 159 PROCEDURE TRIA(X1,Y1,X2,Y2,X3,Y3:INTEGER);
 160 BEGIN
 161  CURSOR(X1,Y1);
 162  DRAW(X2-X1,Y2-Y1);
 163  DRAW(X3-X2,Y3-Y2);
 164  DRAW(X1-X3,Y1-Y3)
 165 END;
 166 PROCEDURE TETRA(X1,Y1,X2,Y2,X3,Y3,X4,Y4:INTEGER);
 167 BEGIN
 168  CURSOR(X1,Y1);
 169  DRAW(X2-X1,Y2-Y1);
 170  DRAW(X3-X2,Y3-Y2);
 171  DRAW(X4-X3,Y4-Y3);
 172  DRAW(X1-X4,Y1-Y4)
 173 END;
 174 PROCEDURE BOX(X1,Y1,X2,Y2:INTEGER);
 175 BEGIN
 176  CURSOR(X1,Y1);
 177  DRAW(0,Y2-Y1);
 178  DRAW(X2-X1,0);
 179  DRAW(0,Y1-Y2);
 180  DRAW(X1-X2,0)
 181 END;
 182 PROCEDURE RANGE(VAR X1,X2:INTEGER; Y:INTEGER);
 183 BEGIN
 184  IF X1>XMIN THEN
 185   REPEAT X1:=X1-1
 186   UNTIL (X1=XMIN) OR POINT0(X1,Y);
 187  IF X2<XMAX THEN
 188   REPEAT X2:=X2+1
 189   UNTIL (X2=XMAX) OR POINT0(X2,Y)
 190 END;
 191 {$S+}
 192 PROCEDURE FILL0(VAR X:INTEGER;Y:INTEGER);
 193 VAR X1,X2:INTEGER;
 194 BEGIN
 195  X2:=X;
 196  RANGE(X,X2,Y);
 197  CURSOR0(X,Y);
 198  X:=X+1;
 199  DRAW0(X2-X,0);
 200  X1:=X;
 201  IF Y<YMAX THEN
 202   WHILE X<X2 DO BEGIN
 203    IF NOT POINT0(X,Y+1) THEN FILL0(X,Y+1);
 204    X:=X+1
 205   END;
 206  X:=X1;
 207  IF Y>YMIN THEN
 208   WHILE X<X2 DO BEGIN
 209    IF NOT POINT0(X,Y-1) THEN FILL0(X,Y-1);
 210    X:=X+1
 211   END;
 212  X:=X2;
 213 END;
 214 {$S-}
 215 PROCEDURE FILL(X,Y:INTEGER);
 216 VAR T,X1:INTEGER;
 217 BEGIN
 218  TRANS(X,Y);
 219  IF NOT POINT0(X,Y) THEN BEGIN
 220   T:=GDTYPE;DTYPE(1);
 221   X1:=X;
 222   FILL0(X1,Y);
 223   DTYPE(T)
 224  END;
 225  CURSOR0(X,Y)
 226 END;
 227 {$S+}
 228 PROCEDURE PAINT0(X,Y,PAT:INTEGER);
 229 VAR X1,X2,PAT1:INTEGER;
 230  FUNCTION ROT(PAT,N:INTEGER):INTEGER;
 231  VAR I:INTEGER;
 232  BEGIN
 233   FOR I:=-1 DOWNTO N DO BEGIN
 234    PAT:=PAT*2;
 235    IF PAT>255 THEN PAT:=PAT-255
 236   END;
 237   FOR I:=1 TO N DO BEGIN
 238    IF PAT MOD 2=1 THEN PAT:=PAT+256;
 239    PAT:=PAT DIV 2
 240   END;
 241   ROT:=PAT
 242  END;
 243 BEGIN
 244  X1:=X;X2:=X;
 245  RANGE(X1,X2,Y);
 246  X1:=X1+1;X2:=X2-1;
 247  PAT:=ROT(PAT,(X-X1) MOD 8);
 248  PAT1:=PAT;
 249  FOR X:=X1 TO X2 DO BEGIN
 250   PAT:=PAT*2;
 251   IF PAT>255 THEN BEGIN
 252    PAT:=PAT-255;
 253    PLOT0(X,Y)
 254   END
 255  END;
 256  IF Y<YMAX THEN BEGIN
 257   PAT:=ROT(PAT1,XSYM*YSYM*GROT);
 258   FOR X:=X1 TO X2 DO BEGIN
 259    IF (PAT>127) AND NOT POINT0(X,Y+1) THEN
PAINT0(X,Y+1,PAT);
 260    PAT:=PAT*2;
 261    IF PAT>255 THEN PAT:=PAT-255
 262   END
 263  END;
 264  IF Y>YMIN THEN BEGIN
 265   PAT:=ROT(PAT1,-XSYM*YSYM*GROT);
 266   FOR X:=X1 TO X2 DO BEGIN
 267    IF (PAT>127) AND NOT POINT0(X,Y-1) THEN
PAINT0(X,Y-1,PAT);
 268    PAT:=PAT*2;
 269    IF PAT>255 THEN PAT:=PAT-255
 270   END
 271  END
 272 END;
 273 {$S-}
 274 PROCEDURE PAINT(X,Y:INTEGER);
 275 VAR T:INTEGER;
 276 BEGIN
 277  TRANS(X,Y);
 278  IF NOT POINT0(X,Y) THEN BEGIN
 279   T:=GDTYPE;DTYPE(1);
 280   PAINT0(X,Y,GPAT);
 281   DTYPE(T)
 282  END;
 283  CURSOR0(X,Y);
 284 END;
 285 PROCEDURE INK(I:INTEGER);
 286 BEGIN
 287  INLINE(55,221,86,2,205,52,34)
 288 END;
 289 PROCEDURE PAPER(P:INTEGER);
 290 BEGIN
 291  INLINE(167,221,86,2,205,52,34)
 292 END;
 293 PROCEDURE BRIGHT(B:INTEGER);
 294 BEGIN
 295  INLINE(151,221,86,2,205,116,34)
 296 END;
 297 PROCEDURE FLASH(F:INTEGER);
 298 BEGIN
 299  INLINE(175,60,221,86,2,205,29,34)
 300 END;
 301 PROCEDURE BORDER(B:INTEGER);
 302 BEGIN
 303  INLINE(221,126,2,205,151,34)
 304 END;
 305 {$L+,O+,C+,S+,I-}
```

Výpis 4. *Program Průběh funkce* (926–V4)

```
PROGRAM PrubehFunkceSinus;
CONST
  PI=3.14159;
VAR
  S,X,U:REAL;

{$F grafR0.W}
{$F grafR1.W}

BEGIN
  GINIT;
  CLS;
  WINDOW(0,0,255,191,16,96);
  AT(2,20);  { pro 64 znaků na řádek }
  WRITELN('PRUBEH FUNKCE Y=SIN(X)');
  S:=90/PI;
  SCALE(S,S);
  LTYPE(3);
  LINE(0,-1,0,1);
  LINE(0,0,2*PI,0);
  LTYPE(1);
  PLOT(0,0);
  U:=3/S;
  X:=0;
  WHILE X<=2*PI DO BEGIN
    LINETO(X,SIN(X));
    X:=X+U
  END
END.
```

Výpis 3. Program Schody (926–V3)

```
PROGRAM SCHODY;
VAR

{$F grafR0.W}
{$F grafR1.W}
{$F grafR2.W}
{$F grafR3.W}
{$F colour.W}

{$L-}
PROCEDURE SCHODY(N:INTEGER);
VAR I:INTEGER;
  PROCEDURE SCHOD;
  BEGIN
    TETRA(0,20,0,10,50,0,50,10);
    DRAW(20,4);
    DRAW(50,-10);
    DRAW(-20,-4);
    FILL(25,10);
    PAINT(35,17)
  END;
BEGIN
  FOR I:=0 TO N-1 DO BEGIN
    SHIFT(20*I,14*I);
    SCHOD
  END;
  SHIFT(0,0);
  IF N>1 THEN BEGIN
    PLOT(50,0);
    DRAW(40,8);
    DRAW(20*(N-2),14*(N-2));
    DRAW(0,20)
  END
END;

{$L+}
BEGIN
  GINIT;
   PAPER(4);
   BORDER(4);
   CLS;
   SCALE(1.3,1.5);
   SCHODY(6);
  WINDOW(24,80,127,159,10,10);
   PAPER(5);
   CLW(TRUE);
   SCALE(0.4,0.4);
   PATTERN(17,-1);
   SCHODY(8);
  WINDOW(96,24,231,111,125,10);
   PAPER(7);
   CLW(TRUE);
   SYMMET(0,1);
   SCALE(0.6,0.6);
   PATTERN(85,1);
   SCHODY(7)
END.
```

Výpis 5. Program Pneumatika (926–V5)

```
PROGRAM PNEUMATIKA;
CONST
  POSUN=15;
  R1=30;
  R2=65;
VAR
  I:INTEGER;
{$F grafI0.W}
{$F grafI1.W}
{$F grafI3.W}

BEGIN
  GINIT;
  CLS;
  COORD(128,88);
  FOR I:=0 TO 1 DO BEGIN
    SHIFT(I*POSUN,I*POSUN);
    CIRCLE(0,0,R1);
    CIRCLE(0,0,R2);
    IF I=0 THEN
      FILL(R1+1,0)
  END
END.
```

SÉRIOVÉ PREPOJENIE MIKROPOČÍTAČA S TLAČIARŇOU

Jozef Kutej a Peter Kottáš, *CHTF SVŠT, Radlinského 9, 812 37 Bratislava*

V mikropočítačoch československej výroby sa stretávame takmer výlučne s paralelným prepojením mikropočítač-tlačiareň. Výhoda, ktorú toto prepojenie má – zvýšená rýchlosť prenosu dát, sa u mikropočítačov neprejaví. U nás bežne používané stolné tlačiarne D 100, PRT 80, K 6313, K 6314 majú rýchlosť tlače do 120 znakov/s. Týmito limitujú maximálnu rýchlosť prenosu dát na úroveň do 2000 bitov/s.

Paralelné prepojenia (systém PMD 85, Centronix, IRPR) vyžadujú na prepojenie minimálne jedenásť až dvanásť vodičov, obvod paralelného styku (aspoň 1/2 8255), budiče zbernice (8286, 8287 . . .) a prípadne ďalšie hradlá na spracovanie riadiacich signálov. Zapojenie je obvodovo náročné, použité konektory, zvyčajne FRB, sú drahé a nedostatkové.

V našich mikropočítačoch sme odskúšali zapojenie sériového interfejsu na prepojenie mikropočítač-tlačiareň. S výhodou sa dá použiť v systémoch s mikroprocesorom MHB 8080A s tromi napájacími hladinami +12 V, +5 V a –5 V.

Interfejs na prepojenie mikropočítača s tlačiarňou vyžaduje tri tranzistory, tri diody a osem odporov. Na prepojenie sú potrebné tri vodiče. Tlačiarne K6313-14, D100, PRT80, môžu byť vybavené odpovedajúcim interfejsom RS 232 (V24), výrobcovia ich ponúkajú.

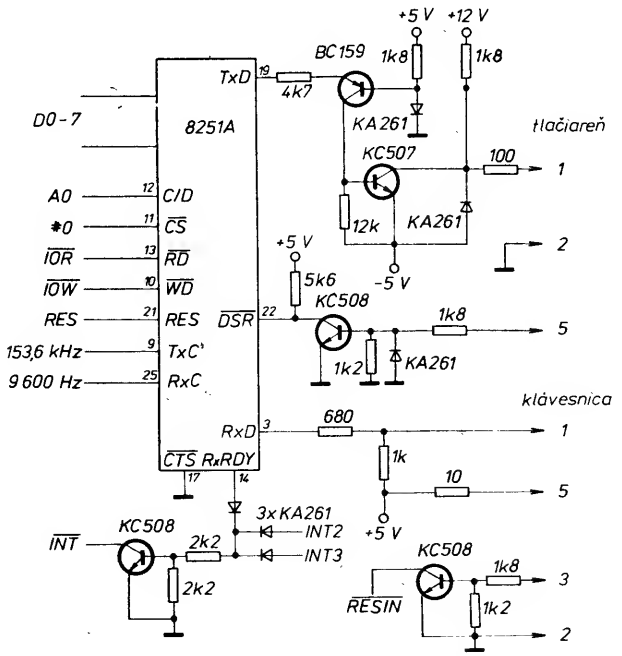
Na **obr. 1** je príklad zapojenia obvodu styku mikropočítača PMD85 s tlačiarňou a klávesnicou. Tlačiareň a klávesnica majú sériové vstupy–výstupy, tlačiareň podľa normy RS232 (V24). Interfejs môže byť ovládaný cez SIO 8251 **(obr. 1),** SIO U856, cez dva vývody PIO 8255, U855 . . .

Výhody zapojenia sú nasledovné:

- redukcia počtu prepojovacích vodičov z jedenásť na tri pri dĺžke prepojovacieho káblu do 16 m,
- zjednodušenie zapojenia obvodu styku, úspora polovodičov,
- na prepojenie nie je potrebné používať drahé a pri častejšom rozoberaní nespoľahlivé konektory FRB. Je možné použiť normalizované 25-kolíkové konektory. No nám sa osvedčili aj lacné 5-kolíkové nf-konektory s mechanickým spojením na závit,
- podstatné zníženie pracnosti pri výrobe prepojovacích káblov.

Tento interfejs je výhodné použiť v mikropočítačových systémoch s napájacími hladinami +12 V, +5 V a –5 V predovšetkým na prepojenie s tlačiarňou. Tu je ekonomicky aj technicky nevhodné realizovať interfejs pomocou IO 75150 a 75154, ktoré vyžadujú na správnu funkciu ďalšiu napájaciu hladinu –12 V.

Zapojenie sme odskúšali pri prepojení mikropočítača PMD85 s tlačiarňami K6311 (NDR) a D100 (PĽR). Na 16 m sme prenášali dáta rýchlosťou 9600 bitov/s.



Obr. 1. Schéma zapojenia vstupov-výstupov pre tlačiareň a klávesnicu. Obvody INT a RESIN umožňujú prácu prerušovacieho systému a externý studený štart z externej klávesnice. (908–1)

SOUŘADNICOVÝ ZAPISOVAČ ŘÍZENÝ MIKROPROCESOREM

Vladimír Julius, OK1IVJ, *Sokolovská 123, 323 16 Plzeň*

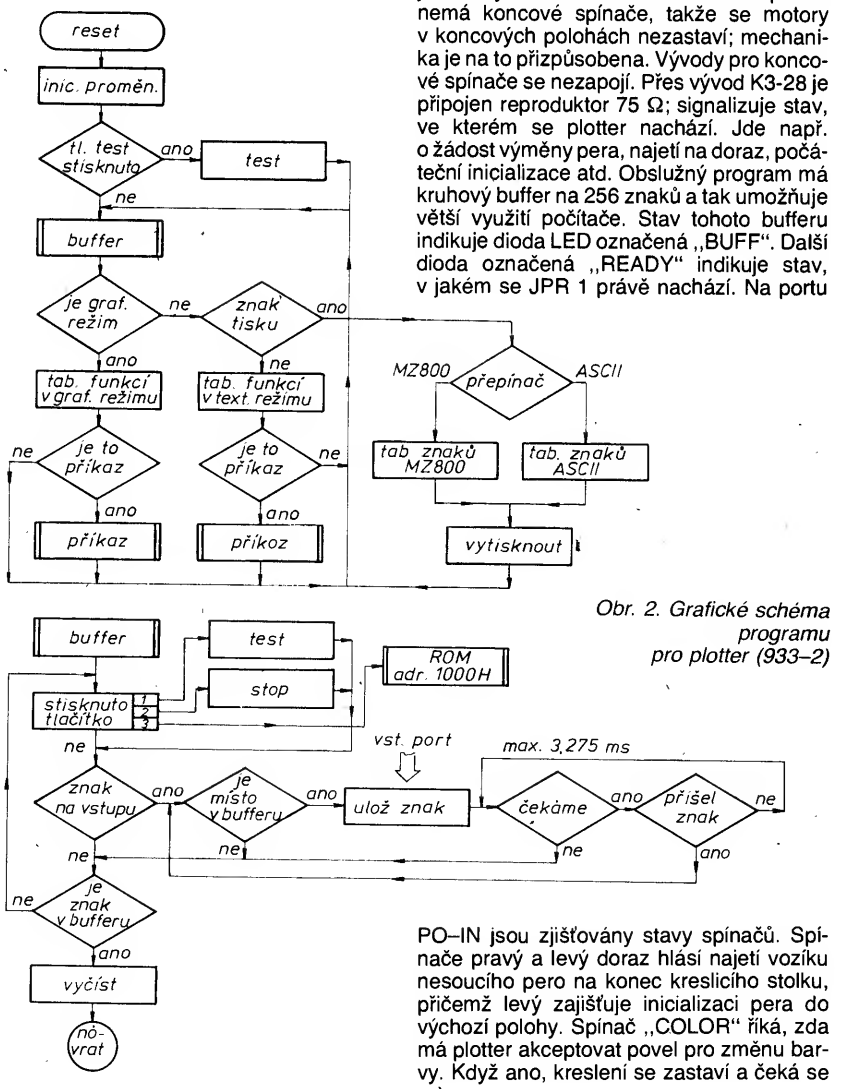
V minulém roce se v naší maloobchodní síti objevily souřadnicové zapisovače 1P16 k počítači SHARP MZ-800. Elektronika zapisovače obsahuje jednočipový mikroprocesor 8050 se 4kB paměti PROM a výstupní zesilovače pro ovládání krokových motorů a pera. Tyto zapisovače používají speciální papír v roli o šířce 11 cm, který není běžně na trhu, a potřeba per v ceně 80 Kčs (souprava) také nepatří k jejich přednostem. Proto jsem zkonstruoval zapisovač, který odstraňuje uvedené nedostatky a umožňuje připojení k libovolnému typu počítače. Používám ho ve spojení s počítačem SHARP MZ – 821 a ZX SPECTRUM +.

V tomto příspěvku nebudu uvádět popis mechanické části zapisovače, její řešení je dnes dostatečně známé a neskrývá při pečlivém provedení žádné záludnosti. Pro posun papíru a pera se všeobecně používají krokové motorky, které se dají jednoduše řídit výstupními porty mikropočítače, jak bude popsáno v dalším textu. Krok zapisovače je převážně volen 0,125 nebo 0,25 mm, což umožňuje mimo jiné kreslení plošných spojů v rastru 2,5 mm. Doporučuji všem případným zájemcům, aby se seznámili s provedením zapisovačů ALFI a MINIGRAF O507.

Řízení zapisovače je řešeno použitím procesorové desky JPR 1. Toto řešení není v dnešní době už jistě tím nejlepším , ale tuto procesorovou desku jsem měl k dispozici a ještě dnes se objevuje v inzerátech AR. Další verze je zpracována a odzkoušena s mikroprocesorem Z80, kde se tak podařilo snížit odběr o více než 50%. Zde jistě mnozí zájemci namítnou, proč jsme nepoužili jednočipový mikropočítač naší produkce např. 8035. Pokud ale budeme počítat, zjistíme, že nelze ušetřit žádné součástky a programování Z80 nám bylo bližší. Jiná situace by byla s použitím μP 8250–4kB ROM . . . Na procesorové desce JPR 1 je osazeno 4 kB EPROM 2716 a 1 kB RAM 2114. Obsazení portů je patrné z **Tab. 1.** Port P1–IN je připojen na řídicí počítač a pomocí bitu RDA a RDP je řízen přenos mezi počítačem a procesorovou deskou JPR 1. Pro větší univerzálnost zařízení je vhodné do řídicích signálů RDA, STA, RDP, IRT zařadit hradla EX-OR MH7486, abychom mohli tyto signály invertovat. Signál STA udává status plotteru a je softwarově ošetřen a nevyužívá se. Signál IRT je použit pro reset JPR 1 od počítače. Port PO–OUT je použit pro řízení motorků a magnetu pro ovládání pera. Tento port je přímo slučitelný se vstupem plotteru MINIGRAF 0507, se kterým byl také odzkoušen. Kdo má možnost použít tento plotter, nemusí si dělat starosti s mechanikou, která je zde vyřešena velice dobře. Tento plotter nemá koncové spínače, takže se motory v koncových polohách nezastaví; mechanika je na to přizpůsobena. Vývody pro koncové spínače se nezapojí. Přes vývod K3-28 je připojen reproduktor 75 Ω; signalizuje stav, ve kterém se plotter nachází. Jde např. o žádost výměny pera, najetí na doraz, počáteční inicializace atd. Obslužný program má kruhový buffer na 256 znaků a tak umožňuje větší využití počítače. Stav tohoto bufferu indikuje dioda LED označená „BUFF". Další dioda označená „READY" indikuje stav, v jakém se JPR 1 právě nachází. Na portu PO–IN jsou zjišťovány stavy spínačů. Spínače pravý a levý doraz hlásí najetí vozíku nesoucího pero na konec kreslicího stolku, přičemž levý zajišťuje inicializaci pera do výchozí polohy. Spínač „COLOR" říká, zda má plotter akceptovat povel pro změnu barvy. Když ano, kreslení se zastaví a čeká se

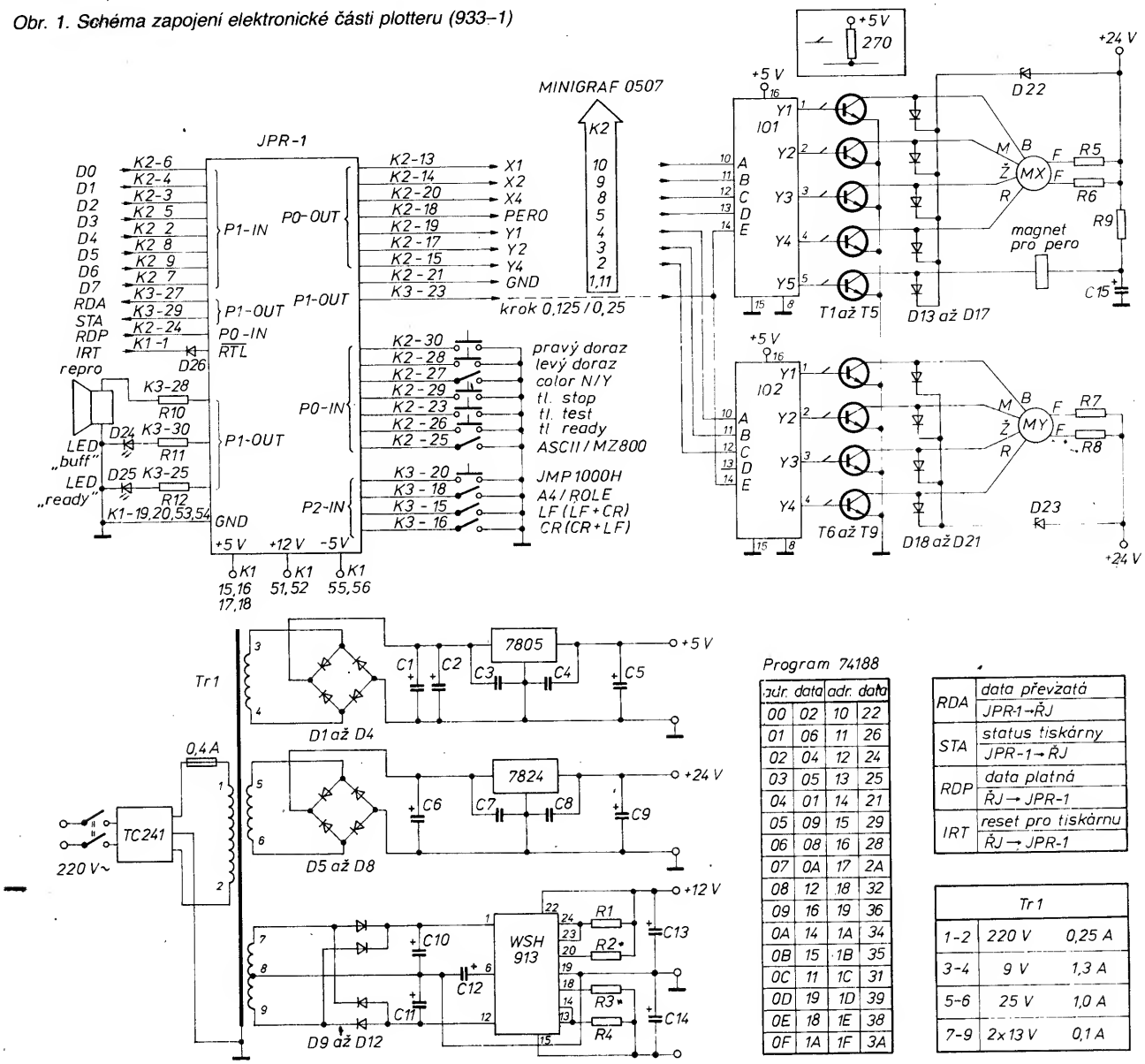
Tab. 1 Obsazení portů JPR - 1

PORT - vst.	bit	KONEKTOR	SIGNÁL H > L
RDSI1	0	K3 - 20	JMP 1000H
	1	K3 - 17	-
	2	K3 - 12	-
	3	K3 - 14	-
	4	K3 - 13	-
	5	K3 - 18	A4 / ROLE
	6	K3 - 15	LF (LF + CR)
	7	K3 - 16	CR (CR + LF)
RDSI	0	K2 - 30	PRAVÝ DORAZ
	1	K2 - 28	LEVÝ DORAZ
	2	K2 - 27	COLOR A/N
	3	K2 - 29	TL. STOP
	4	K2 - 23	TL. TEST
	5	K2 - 26	TL. READY
	6	K2 - 24	RDP (DATA PLATNÁ)
	7	K2 - 25	ASCII / MZ 800
DSHI	0	K2 - 6	DATA 0
	1	K2 - 4	1
	2	K2 - 3	2
	3	K2 - 5	3
	4	K2 - 2	4
	5	K2 - 8	5
	6	K2 - 9	6
	7	K2 - 7	7
PORT - výst.	**bit**	**KONEKTOR**	**SIGNÁL**
RDSO	0	K3 - 24	-
	1	K3 - 26	-
	2	K3 - 27	RDA (DATA AKCEPT)
	3	K3 - 29	STA (STATUS TISK.)
	4	K3 - 23	KROK 0.125/0.25
	5	K3 - 28	REPRODUKTOR
	6	K3 - 30	LED "BUFF"
	7	K3 - 25	LED "READY"
MOTR	0	K2 - 13	FÁZE - X 1
	1	K2 - 14	X 2
	2	K2 - 20	X 4
	3	K2 - 19	Y 1
	4	K2 - 17	Y 2
	5	K2 - 15	Y 4
	6	K2 - 16	-
	7	K2 - 18	OVL. PERA



Obr. 2. Grafické schéma programu pro plotter (933–2)

Obr. 1. Schéma zapojení elektronické části plotteru (933–1)



Program 74188

adr.	data	adr.	data
00	02	10	22
01	06	11	26
02	04	12	24
03	05	13	25
04	01	14	21
05	09	15	29
06	08	16	28
07	0A	17	2A
08	12	18	32
09	16	19	36
0A	14	1A	34
0B	15	1B	35
0C	11	1C	31
0D	19	1D	39
0E	18	1E	38
0F	1A	1F	3A

RDA	data převzatá
	JPR-1 → ŘJ
STA	status tiskárny
	JPR-1 → ŘJ
RDP	data platná
	ŘJ → JPR-1
IRT	reset pro tiskárnu
	ŘJ → JPR-1

Tr1		
1-2	220 V	0,25 A
3-4	9 V	1,3 A
5-6	25 V	1,0 A
7-9	2x13 V	0,1 A

na výměnu pera. Z reproduktoru se ozývá přerušovaný tón signalizující barvu pera, kterou si počítač žádá. Jde o sled 1–4 teček. Stiskem tlačítka „READY" se plotter opět spustí. Při rozepnutí spínače „COLOR" máme možnost kreslit barevný obrazec pouze jednou barvou bez přerušení. Tlačítko „STOP" umožňuje okamžité zastavení plotteru – spustí se opět tlačítkem „READY". Protože tento plotter byl řešen hlavně pro použití s počítačem SHARP, který má pro malá písmena abecedy jiné kódy než udává tabulka ASCII, je nutné tyto znaky převádět. To je zajištěno s přepínačem přepnutým do polohy MZ-800. Pokud je tento přepínač v poloze ASCII, znaky se tisknou podle této tabulky. Znaky, které nejsou v generátoru znaků obsaženy, jsou nahrazeny tečkou. Tlačítko „JMP 1000H" vykoná tuto instrukci a provede nepodmíněný skok na adresu 1000 pro případné další využití JPR1. Přepínač „A4/ROLE" sdělí počítači, jestli máme založen list papíru A4 a pro další psaní musíme inicializovat znovu začátek stránky, nebo zda máme roli a stačí vynechat několik volných řádků. K přepínačům „LF" a „CR" jistě není co dodávat. Ještě je třeba se zmínit o výstupu pro velikost kroku 0,125 nebo 0,25 mm. Tento výstup vybírá dekodér naprogramovaný v paměti MH74188. Pokud bude mít někdo připojen plotter 0507, potom nemůže tuto funkci využít a vývod zůstane nezapojen.

Ovládání krokových motorů je řízeno tříbitovým kódem 0–7, který přes dekodér ovládá 4 fáze krokového motoru. Tento dekodér v režimu 0,125 mm otáčí motorem o 4,5 stupně, což představuje 80 kroků na jednu otáčku. V režimu 0,25 mm se motor otáčí o 9 stupňů na jeden krok. Úměrně tomu je také softwarově zmenšena rychlost krokování tak, aby výsledná rychlost psaní zůstala nezměněna. To platí při použití dekodéru MH74188, který je pro tento účel naprogramován podle tabulky. Řešení s dekodérem bylo použito vzhledem k jednoduchému připojení na plotter MINIGRAF 0507, který obdobný dekodér obsahuje. V další verzi programu je tento dekodér řešen softwarově, což dále zjednodušuje elektroniku plotteru.

V plotteru jsou použity motory SMR 300/300 RI24, které se dají objednat pro organizaci v MEZ Náchod. V maloobchodní síti se dají občas koupit motory SMR 300/100, které pracují stejně, pouze výsledná rychlost se bude muset v programu zmenšit asi o 1/3.

Dále popsaný program řadí tento plotter mezi inteligentní periférie, které na našem trhu stále nejsou a zřejmě ještě dlouho nebudou v takových cenových relacích, aby byly dostupné uživatelům domácích počítačů. Máme tak vyřešen grafický výstup počítače a jsme schopni kreslit technické výkresy, výkresy plošných spojů a třeba diplomy nebo přání k svátku.

Seznam součástek

Rezistory:

R1, R4	6,8 Ω	TR 191
R2*	12 kΩ	TR 191
R3*	8,2 kΩ	TR 191
R5-R8	68 Ω	TR 510
R9	47 Ω	TR 510
R10-R12	100 Ω	TR 191

Kondenzátory:

C1, C2, C5	1 GF	TE 984
C3, C4, C7, C8	100 nF	TK 782
C6, C15	500 µF	TE 986
C9	50 µF	TE 986
C10, C11	200 µF	TE 986
C12-C14	4,7 µF	TE 125

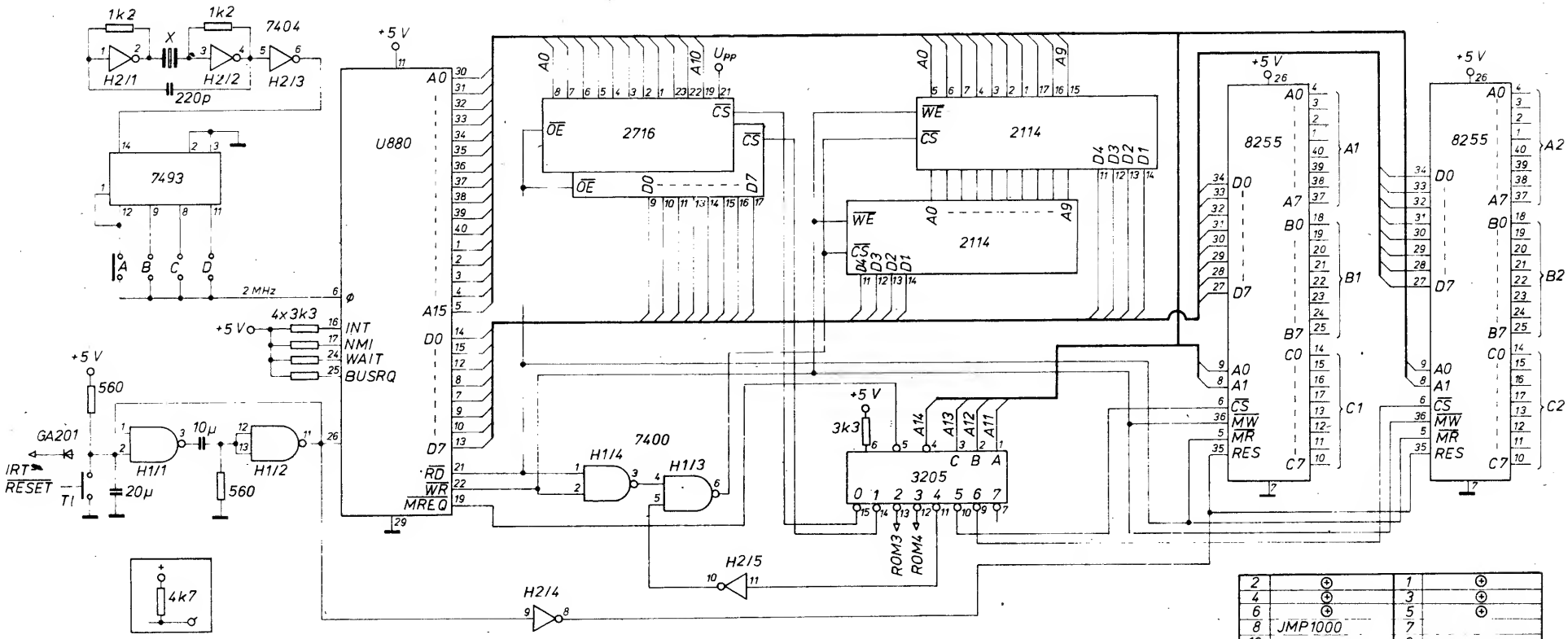
Polovodiče:

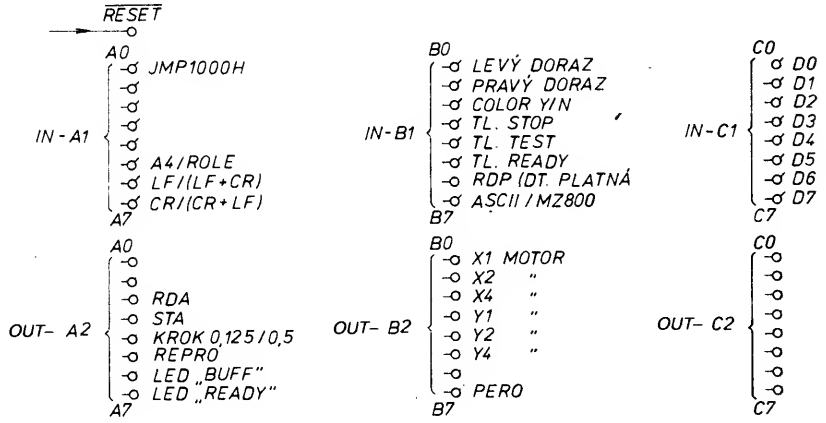
D1-D8	1N5401
D9-D12	KY130/80
D13-D21	KY130/150
D22, D23	KZ260/18
D24	LQ1702
D25	LQ1202
D26	GA204
T1-T9	KF508
I01, I02	MH74188
I03	MH7805
I04	MH7824
I05	WSH913

Ostatní:

MX, MY	SMR 300/300 RI24
REPRO ARZ 084	

* Nutno nastavit výstupní napětí.





ADRESA	
0000 - 07FF	ROM1
0800 - 0FFF	ROM2
1000 - 17FF	RES. (ROM3)
1800 - 1FFF	RES. (ROM4)
2000 - 27FF	RAM
2800 - 2FFF	PIO1
3000 - 37FF	PIO2

2	⊕	1	⊕
4	⊕	3	⊕
6	⊕	5	⊕
8	JMP1000	7	
10		9	
12		11	RDA
14		13	STA
16		15	
18	A4/ROLE	17	REPRO
20	LF(LF+CR)	19	LED „BUFF"
22	CR(CR+LF)	21	LED „READY"
24	LEVÝ DORAZ	23	MOTOR X1
26	PRAVÝ DORAZ	25	" X2
28	COLOR Y/N	27	" X4
30	TL. STOP	29	" Y1
32	TL. TEST	31	" Y2
34	TL. READY	33	" Y4
36	RDP	35	
38	ASCII / MZ800	37	PERO
40	D0	39	
42	D1	41	
44	D2	43	
46	D3	45	
48	D4	47	
50	D5	49	
52	D6	51	
54	D7	53	
56	IRT	55	
58	⊖	57	⊖
60	⊖	59	⊖
62	⊖	61	⊖

KONEKTOR FRB

Obr. 3. Náhrada desky JPR-1 zapojením s mikroprocesorem Z80 (933–3)

```
;*****************************************************
;*                                                   *
;*  Program pro rizeni plotteru z ridici jednotky    *
;*       pres standartni rozhrani Centronics         *
;*                                                   *
;*   Julius Roman  Sokolovska 123  Plzen 323 16      *
;*                                                   *
;*****************************************************



.8080                   ;soubor instrukci I8080
.RADIX 16               ;implicitni ciselna soustava

SYKORA  EQU     1000    ;zacatek dalsich 4 kbyte
NEWP    EQU     8000    ;zde zacina vlastni program
                        ;JPR-1 EQU 0
RAM     EQU     08000   ;pocatecni adresa promennych
                        ;JPR-1 EQU 2000
RDSI1   EQU     8E      ;vstup ridicich signalu
                        ;JPR-1 EQU 2C00
RDSI    EQU     90      ;vstup ridicich signalu
                        ;JPR-1 EQU 2400
RDSO    EQU     8C      ;vystup ridicich signalu
                        ;JPR-1 EQU 2800
DSHI    EQU     8D      ;vstup dat
                        ;JPR-1 EQU 2800
MOTR    EQU     90      ;vystup na krokove motory
                        ;JPR-1 EQU 2400

        .PHASE  NEWP
        DI              ;zakaz preruseni
        LXI     SP,STACK;a nastav stack
        MVI     A,4     ;pocatecni nastaveni
        STA     RDSO    ;vystupniho portu
        XRA     A       ;znuluj
        STA     KLM     ;bit reproduktoru
        STA     PPORG   ;funkci dorazu pro ORG
        STA     KR025   ;krok= 0.125 mm
        LDA     RDSI    ;otestuj tlacitko
        ANI     10      ;TEST, je-li stisknuto
        CZ      TEST    ;proved celkovy test
        CALL    MUS1    ;piskej
        CALL    GENINI  ;inicializace generatoru

RS1:    XRA     A       ;znuluj
        STA     KLM     ;bit reproduktoru
        STA     UKRA    ;ukazatel radek
        STA     UKZR    ; - " -   znaku na radce
        STA     KR025   ;jednoduchy krok 0.125 mm
        STA     RDSO    ;ridici signaly
        STA     SPD     ;rychlost
        STA     PPORG   ;funkci dorazu pro ORG
        INR     A       ;barva=1
        STA     COLR
        MVI     A,3F    ;pocet radek na stranku
        STA     PRST
        CALL    BUFINI  ;inicializace bufferu
        LXI     B,0
        LXI     D,0     ;parametry pro ORG
        CALL    PORG    ;nastav pocatek sour. syst.
        CALL    CH400   ;80 znaku na radek
        LXI     H,0     ;skutecna X=0
        SHLD    SKUTX
        MVI     A,1
        STA     REZI    ;textovy rezim
        STA     ASCD    ;rezim ASCII
        STA     PPORG   ;funkce dorazu pro havarii

RIBL:   CALL    BUFE    ;cyklus pro vycitani prikazu
        LXI     D,RIBL  ;navratova adresa pro pprog.
        PUSH    D       ;do stacku
        LDA     REZI
        CPI     2       ;podle druhu rezimu vyber tab.
        LXI     H,TBFG  ;graficky-graficka tabulka
        JZ      RIBG    ;a vyber
        MOV     A,C     ;rezim textovy
        CPI     1D      ;je-li kod 1DH
        JZ      RIB1    ;vyber adresu
        ANI     0F0     ;je-li vetsi nez 0FH ->
        JNZ     PRINT   ;tisknutelny znak - vytiskni

RIB1:   LDA     RDSI    ;ridici znak
        ANI     80
        CZ      PREVOD  ;preved podle prepinace
        LXI     H,TBFT  ;textova tabulka

RIBG:   CALL    PROHL   ;vyber podprogram z tabulky
        PCHL            ;a skoc na nej

PROHL:  MOV     A,M     ;v C je povel, v HL adresa tab.
        CMP     C       ;vybere adresu
        JZ      PROHU   ;nasel prikaz
        CPI     0FF     ;konec tabulky
        JZ      PROHK
        INX     H       ;HL=HL+3 ->
        INX     H       ;dalsi pozice
        INX     H
        JMP     PROHL   ;dalsi prikaz v tabulce

PROHU:  INX     H       ;vybere z tabulky adresu
        MOV     E,M     ;do E
        INX     H
        MOV     D,M     ;dalsi pulku do D
        XCHG            ;a vymen s HL
        RET             ;navrat

PROHK:  POP     H       ;povel jsem nenasel
        POP     H       ;zrus navratovou adresu
        JMP     RIBL    ;dalsi povel

HAVAR:  LXI     SP,STACK;zjistil jsem doraz
        CALL    MUS2    ;a hlasim havarii
        JMP     RS1     ;restartuj system

TBFG:   DB      48      ;tabulka funkci
        DW      HOME    ;v grafickem rezimu
        DB      54
        DW      SETS
        DB      45
        DW      CGRAMS
        DB      47
        DW      CGRAMW
        DB      43
        DW      COLOR
        DB      4A
        DW      RLINE
        DB      52
        DW      RMOVE
        DB      4C
        DW      LTYPE
        DB      44
        DW      DRAW
        DB      49
        DW      HSET
        DB      4D
        DW      PMOVE
        DB      58
        DW      AXIS
        DB      53
        DW      SCALE
        DB      51
        DW      ANGLE
        DB      50
        DW      GPRINT
        DB      1
        DW      TEXTR
        DB      46
        DW      FAST
        DB      5A
        DW      SSIZE
        DB      59
        DW      DOT
        DB      57
        DW      RDOT
        DB      42
        DW      DIAC
        DB      0FF
        DW      PROHK

TBFT:   DB      2       ;funkce v textovem rezimu
        DW      GRAFR
        DB      5
        DW      CGRAMW
        DB      0D
        DW      PCR
        DB      0A
        DW      PLF
        DB      8
        DW      BASP
        DB      3
        DW      LINUP
        DB      4
        DW      PTEST
        DB      9
        DW      DEVET
        DB      0B
        DW      CH26
        DB      7
        DW      CH408
        DB      1
        DW      TEXTT
        DB      0C
        DW      STRAN
        DB      1D
        DW      CHCOL
        DB      0FF
        DW      PROHK

BUFINI: XRA     A       ;inicializace bufferu
        LXI     H,CW    ;znuluj 1.ukazovatko
        MOV     M,A
        INX     H
        MOV     M,A     ;pote i druhe
        STA     CK
        STA     BUFINT
        RET

BUFE:   PUSH    PSW     ;pprog. pro rizeni prijmu
```

```
        PUSH    D       ;dat a prace s bufferem
        PUSH    H
        LDA     RDSI    ;test tl. STOP
        ANI     8
        CZ      MUS3    ;stisknuto-pozdrz praci
        LDA     RDSI    ;test tl. TEST
        ANI     10
        CZ      PTEST   ;stisknuto-proved test
        LDA     RDSI1   ;test tl. SYKORA
        ANI     1
        JZ      SYKORA  ;stisknuto-skoc na jeho adr.
        CALL    BUFO    ;ulozeni znaku z portu
        LDA     BUFINT  ;znak cten z buff.
        ORA     A       ;nebo z pomocne promenne
        JNZ     NEZB    ;-z promenne
        CALL    BFI     ;-z bufferu
        PUSH    PSW     ;uschovej
        LDA     KR025
        ORI     40
        STA     RDSO    ;rozsvit diodu READY
        POP     PSW
        JZ      BUFE+3  ;buff. prazdny - LED sviti
        LDA     KR025
        STA     RDSO    ;neni, zhasni ji
BUFZ:   POP     H
        POP     D
        POP     PSW
        RET             ;obnov reg. a vrat se

NEZB:   XRA     A       ;znak neni z bufferu
        STA     BUFINT  ;priste uz bude
        LDA     PRECH   ;nacti jej do C
        MOV     C,A
        JMP     BUFZ    ;a vrat se

BUFO:   LDA     RDSI    ;jsou pripravena data ?
        ANI     40
        RZ              ;ne, vrat se
        LDA     DSHI    ;ano, cti je do C
        MOV     C,A
        CALL    BFO     ;pokus se je ulozit do buff.
        RZ              ;nejde to - je plny
        LDA     KR025
        ORI     4
        STA     RDSO    ;potvrd prijem data
SEST:   LDA     RDSI    ;cekej na shozeni RDP
        ANI     40
        JNZ     SEST
        LDA     KR025
        STA     RDSO    ;nastav znovu pripravenost
        MVI     B,83
BUFW:   DCR     B       ;chvili pockej
        RZ              ;nejdou, vratime se
        LDA     RDSI    ;a testuj prijdou-li
        ANI     40      ;dalsi data
        JNZ     BUFO    ;prisli
        JMP     BUFW    ;cekej dale

BFI:    LXI     H,CK    ;vlastni cteni z bufferu
        MOV     A,M
        RAR
        XRA     M
        ANI     1
        LXI     H,BUFF-2
        MOV     A,M
        MOV     E,A
        INX     H
        JNZ     BFI2
        CMP     M
        RZ
BFI2:   MVI     D,0
        XCHG
        INX     H
        DAD     D
        MOV     C,M
        XCHG
        DCX     H
        INR     M
        RNZ
        LXI     H,CK
        MOV     A,M
        RAR
        RAR
        CMC
        RAL
        RAL
        JMP     BFO3

BFO:    LXI     H,CK    ;vlastni zapis do bufferu
        MOV     A,M
        RAR
        XRA     M
        ANI     1
        LXI     H,BUFF-2
        MOV     A,M
        INX     H
        JZ      BFO2
        CMP     M
        RZ
BFO2:   MVI     D,0
        MOV     E,M
        XCHG
        INX     H
        DAD     D
        MOV     M,C
        XCHG
        INR     M
        RNZ
        LXI     H,CK
        MOV     A,M
        RAR
        CMC
        RAL
BFO3:   MOV     M,A
        INR     A
        RET

PARAM:  CALL    IGNO    ;precteni ciselnych param.
        MOV     A,C     ;z bufferu
        CPI     2D      ;test na zaporna cisla
        MVI     A,0     ;kladne
        JNZ     KL1
        MVI     A,1     ;zaporne
KLAD:   STA     PRZ     ;nastav priznak pro pozdejsi
        LXI     H,0     ;invertovani
        CALL    EX0     ;vlastni cteni
        JNC     COMP    ;zinvertovani podle znamenka
        MOV     A,C     ;uloz posledni znak znovu
        STA     PRECH   ;do bufferu
        MVI     A,1
        STA     BUFINT

COMP:   LDA     PRZ     ;je-li tato prom. nastavena
        CPI     1       ;na 1 neprovadi invertaci HL
        RNZ
        MOV     A,L     ;nejprve L
        CMA
        MOV     L,A
        MOV     A,H     ;pote i H
        CMA
        MOV     H,A
        INX     H       ;HL=HL+1
        RET             ;zinvertovane HL

KL1:    MVI     A,1     ;neni-li 1. nemezerovy znak
        STA     BUFINT  ;"-" uloz jej zpet do buff.
        MOV     A,C
        STA     PRECH
        JMP     KLAD    ;cislo je kladne

EX0:    CALL    BUFE    ;vlastni cteni vstup HL=0
        MOV     A,C     ;vystup cislo v HL
        CPI     3A
        CMC
        RC              ;znak mimo rozsah vrat se
EX1:    MOV     B,A
        CALL    NIBBLE
        RC
        DAD     H       ;zvys HL o dalsi cislo
        PUSH    H
        DAD     H
        DAD     H
        POP     D
        DAD     D
        MOV     E,A
        MVI     D,0
        DAD     D
        JMP     EX0

NIBBLE: SUI     30      ;testuj prislusnost
        RC              ;do mnoziny
        ADI     30-47   ;hexadecimalnich znaku
        RC              ;nepatri tam
        ADI     6
        JP      NI0
        ADI     7
        RC              ;nepatri tam
NI0:    ADI     0A
        ORA     A
        RET

IGNO:   CALL    BUFE    ;ignoruj mezery
        MVI     A,20
        CMP     C
        JZ      IGNO
        RET

HSET:   LHLD    SKUTX   ;funkce HSET
        XCHG
        LHLD    STPX    ;nastav novy skut. pocatek
        DAD     D
        SHLD    SKUTX
        JMP     NULXY   ;a nuluj okamzite X a Y

HOME:   LXI     B,0     ;funkce HOME
        LXI     D,0
        JMP     MOVA    ;presun na 0,0

COLOR:  CALL    PARAM   ;funkce COLOR
        INR     L
```

```
        MOV     A,L
COL1:   STA     CKLO
        STA     COLR
        LDA     RDSI
        ANI     4
        RNZ             ;barva je zakazana, vrat se
        LXI     H,0B0   ;neni, inicializuj zvukovy
        SHLD    KLOPA   ;generator
        LXI     H,800   ;podle cisla barvy
        SHLD    KLOPB
        MVI     A,15
        STA     KMIT
        JMP     MUSIC   ;hlas barvu

;Umoznuje nadefinovat vlastni znak

CGRAMS: CALL    PARAM
        MOV     A,L     ;cislo generatoru
        ORA     A
        JNZ     CGRA
        INR     A
CGRA:   ANI     0F      ;uvazuje pouze 15 generatoru
        LXI     H,CGR-22;adresa generatoru
        LXI     D,22    ;maximalni delka
        CALL    NASOB   ;vypocit vyslednou adr.
        PUSH    H       ;uschovej ji
        MVI     E,1F    ;pocet znaku v generatoru
        INX     H       ;vyhrad byte pro delku
        XRA     A       ;uvodni byte nastav
        MOV     M,A     ;na nulu
        INX     H       ;dalsi pozice
        CALL    BUFE    ;oddelovac pryc
CGRS1:  CALL    BUFE    ;znak z bufferu
        MOV     A,C
        MOV     M,A     ;do generatoru
        INX     HL      ;dalsi pozice
        DCR     E       ;sniz zbyvajici pocet
        JZ      CGRSK   ;znaku, je-li 0 ukonci to
        ANI     80      ;je-li 7.bit nastaven
        JZ      CGRS1   ;konec tvoreni znaku
        MVI     A,20    ;vypocti delku
        SUB     E
        POP     H       ;a uloz ji
        MOV     M,A
        RET
CGRSK:  ORI     80      ;vycerpana povolena delka
        DCX     HL      ;nastav 7.bit na jednicku
        MOV     M,A     ;uloz to
        MVI     A,21    ;nastav maximalni
        POP     H       ;delku
        MOV     M,A
        RET             ;navrat

;Vytisknuti preddefinovaneho znaku

CGRAMW: CALL    PARAM   ;cislo generatoru
        MOV     A,L
        ORA     A
        JNZ     CGRC
        INR     A
CGRC:   ANI     0F      ;uvazuje pouze 15 generatoru
        LXI     H,CGR-22;adresa generatoru
        LXI     D,22
        CALL    NASOB   ;vypocti adresu
        MOV     A,M     ;testuj delku
        ORA     A
        RZ              ;vrat se je-li 0
        MOV     E,A
        INX     H
        JMP     WRITE   ;jdi tisknout

GENINI: LXI     H,CGR   ;inicializuje generator
        LXI     D,1FF   ;v delce 1/2 kbyte
GENIN1: XRA     A
        MOV     M,A
        INX     H
        DCX     D
        MOV     A,E
        ORA     D
        JNZ     GENIN1  ;pokracuj
        RET             ;hotovo

PMOVE:  CALL    SXY     ;funkce PMOVE
        JMP     MOVA

DOT:    CALL    SXY     ;funkce POINT
        JMP     POINTA

RDOT:   CALL    SXY     ;funkce RPOINT
        JMP     POINTR

SSIZE:  CALL    PARAM   ;funkce SIZE
        SHLD    SIXX    ;XX
        CALL    BUFE
        CALL    PARAM
        SHLD    SIXY    ;XY
        CALL    BUFE
        CALL    PARAM
        SHLD    SIYX    ;YX
        CALL    BUFE
        CALL    PARAM
        SHLD    SIYY    ;YY
        RET

DRAW:   CALL    RYCH    ;funkce DRAW
        CALL    SXY
        CALL    VECTA
        JMP     RYCHN

LTYPE:  CALL    PARAM   ;funkce LTYPE
        MOV     A,L
        RLC
        RLC             ;parametr nasob 4
        STA     TPLI    ;uloz jej
        STA     AKLI
        STA     RTLI
        MVI     A,80
        STA     PSLI
        STA     STWP    ;inicializuj promenne
        RET             ;pro typ cary

SETS:   CALL    PARAM   ;funkce SPEED
        MOV     A,L
        STA     SPD     ;uloz rychlost
        JMP     RYCHN

RYCH:   LDA     RTLI    ;prepocte rychlost
        CPI     0       ;na skutecnou
        JZ      RYCHN   ;podle delky kroku
        STA     TPLI    ;a typu cary
        LDA     STWP
        STA     PSLI
        MVI     B,18
        CALL    SPEDA
        RET
RYCHN:  MVI     B,0B    ;nastav rychlost
        CALL    SPEDA   ;pro presuny
        XRA     A
        STA     TPLI
        LDA     PSLI
        STA     STWP
        MVI     A,80
        STA     PSLI
        RET
SPEDA:  LDA     SPD
        ADD     B
        STA     SPED
        RET

RMOVE:  CALL    SXY     ;funkce RMOVE
        JMP     MOVR

RLINE:  CALL    RYCH    ;funkce RLINE
        CALL    SXY
        CALL    VECTR
        JMP     RYCHN

SXY:    CALL    PARAM   ;nacte z bufferu
        PUSH    H       ;souradnice X a Y
        CALL    BUFE
        CALL    PARAM
        MOV     B,H     ;do BC
        MOV     C,L
        POP     D       ;a DE
        RET

AXIS:   CALL    BUFE    ;funkce AXIS
        MOV     A,C
        CPI     31      ;osa do smeru
        JZ      AXISX   ;X
        CALL    AXIA    ;parametry osy
DILY:   LXI     B,0
        LXI     D,0FFFC
        CALL    VECTR   ;dilek
        LXI     B,0
        LXI     D,8
        CALL    VECTR
        LXI     D,0FFFC
        LXI     B,0
        CALL    VECTR
        LXI     D,0
        LHLD    PITCH
        MOV     B,H
        MOV     C,L
        CALL    VECTR   ;na dalsi pozici
        LDA     REPE    ;sniz pocet opakovani
        DCR     A
        JZ      AXYK    ;konec osy
        STA     REPE    ;jeste ne
        JMP     DILY
AXISX:  CALL    AXIA    ;parametry osy
DILX:   LXI     B,0FFFC ;dilek
        LXI     D,0
        CALL    VECTR
        LXI     B,8
        LXI     D,0
        CALL    VECTR
        LXI     B,0FFFC
        LXI     D,0
        CALL    VECTR
        LXI     B,0
```

```
        LHLD    PITCH
        XCHG
        CALL    VECTR   ;na dalsi pozici
        LDA     REPE    ;sniz pocet opakovani
        DCR     A
        JZ      AXXK    ;konec osy
        STA     REPE    ;jeste ne
        JMP     DILX

AXIA:   CALL    BUFE    ;nacte parametry osy
        CALL    PARAM
        SHLD    PITCH   ;roztec
        CALL    BUFE
        CALL    PARAM
        MOV     A,L
        STA     REPE    ;pocet dilku
        RET

AXXK:   LXI     D,0     ;posledni dilek
        LXI     B,0FFFC
        CALL    VECTR
        LXI     D,0
        LXI     B,8
        CALL    VECTR
        LXI     D,0
        LXI     B,0FFFC
        JMP     VECTR
AXYK:   LXI     B,0
        LXI     D,0FFFC
        CALL    VECTR
        LXI     B,0
        LXI     D,8
        CALL    VECTR
        LXI     B,0
        LXI     D,0FFFC
        JMP     VECTR

GPRINT: CALL    BUFE    ;funkce GPRINT
        MOV     A,C
        CPI     0D      ;je-li CR znamena to konec
        RZ              ;retezce
        LDA     RDSI
        ANI     80
        CZ      PREVOD  ;prevede znak podle prep.
        MOV     A,C
        ANI     80
        MOV     A,C     ;je-li stalem vetsi nez 127
        JNZ     GTEC    ;vytisken tecku
        CALL    PCHR    ;vytiskni znak
        JMP     GPRINT  ;znovu
GTEC:   MVI     A,2E    ;tecka
        CALL    PCHR    ;vytiskni
        JMP     GPRINT  ;znovu

NASOB:  DAD     D       ;A krat pricte DE k HL
        DCR     A
        JNZ     NASOB
        RET

ANGLE:  CALL    PARAM   ;funkce ANGLE
        MOV     A,L     ;uhel do A
        CPI     0
        JZ      VODP    ;vodorovne vpravo
        CPI     1
        JZ      DOL     ;svisle dolu
        CPI     2
        JZ      VODL    ;vodorovne vlevo
        CPI     3
        JZ      NAH     ;svisle nahoru
        RET             ;neznam
VODP:   CALL    VISI
        SHLD    SIXX    ;XX,YY = velikost
        SHLD    SIYY
        LXI     H,0
        SHLD    SIXY
        SHLD    SIYX    ;XY,YX = 0
        RET
DOL:    CALL    VISI
        SHLD    SIYX    ;YX = velikost
        MVI     A,1
        STA     PRZ
        CALL    COMP
        SHLD    SIXY    ;XY = -velikost
        LXI     H,0
        SHLD    SIXX
        SHLD    SIYY    ;XX,YY = 0
        RET
NAH:    CALL    VISI
        SHLD    SIXY    ;XY = velikost
        MVI     A,1
        STA     PRZ
        CALL    COMP
        SHLD    SIYX    ;YX = -velikost
        LXI     H,0
        SHLD    SIXX
        SHLD    SIYY    ;XX,YY = 0
        RET

VODL:
        CALL    VISI
        MVI     A,1
        STA     PRZ
        CALL    COMP
        SHLD    SIXX
        SHLD    SIYY    ;XX,YY = -velikost
        LXI     H,0
        SHLD    SIYX
        SHLD    SIXY    ;XY,YX = 0
        RET
VISI:   LDA     SIZE    ;podle SIZE vypocte
        INR     A       ;do HL skutecnou velikost
        MVI     H,0
        MOV     L,A
        RET
DIAC:   CALL    PARAM   ;nastavi nebo zrusi
        MOV     A,L     ;diakriticky rezim
        STA     ASCD
        RET
SCALE:  CALL    PARAM   ;nastavi velikost znaku
        MOV     A,L
        STA     SIZE
        MVI     A,0     ;a uhel tisku 0
        JMP     ANGLE+4
PTEST:  LDA     REZI    ;neni-li nastaven textovy
        CPI     1       ;rezim nastav ho
        CNZ     TEXTR
        CALL    CR
        CALL    LF
        CALL    CH26
        LXI     H,TSTTB ;vytiskni text
        MVI     A,0E    ;podle tabulky
TSTT:   PUSH    PSW
        PUSH    H
        MOV     C,M
        CALL    PRINT
        POP     H
        INX     H
        POP     PSW
        DCR     A
        JNZ     TSTT
        CALL    CR
        CALL    LF
        CALL    MONOP
        MVI     A,4
PTES2:  PUSH    PSW     ;a podtrhni ho
        LXI     B,0
        LXI     D,1EF
        CALL    VECTR
        LXI     B,0FFFE
        LXI     D,0FE11
        CALL    MOVR
        POP     PSW
        DCR     A
        JNZ     PTES2
        CALL    LF
        JMP     TEXTT
GRAFR:  MVI     L,1     ;velikost znaku 1
        CALL    SCALE+3
        MVI     L,0     ;typ cary 0
        CALL    LTYPE+3
        MVI     A,2     ;graficky rezim
        STA     REZI
        CALL    MONOP   ;presun na zacatek
        LHLD    STPX    ;inicializuj
        SHLD    SKUTX   ;souradnice
        CALL    NULXY
        RET
TEXTT:  CALL    CH408
        JMP     CR

TEXTR:  MVI     L,0     ;typ cary 0
        CALL    LTYPE+3
        MVI     A,1     ;textovy rezim
        STA     REZI
        STA     ASCD    ;ASCII rezim
        LHLD    SKUTX
        XCHG
        LHLD    STPX
        DAD     D
        SHLD    STPX    ;vypocti skutecnou X
        LXI     H,0
        SHLD    NOPX
        SHLD    SIXY
        SHLD    SIYX    ;parametry pro znak
        CALL    CH408   ;80 znaku na radek
        XRA     A
        STA     UKRA
        STA     UKZR    ;znuluj ukazatele
        CALL    FAST+4  ;jednoduchy krok
        RET
DEVET:  CALL    BUFE    ;prisel kod $09
        MOV     A,C     ;cti dalsi znak
        CPI     9
        JZ      DEV1    ;je to opet $09
        STA     PRECH   ;neni - vrat jej
        MVI     A,1     ;do bufferu
        STA     BUFINT
        RET             ;a zpet
DEV1:   CALL    BUFE    ;dalsi znak
        MOV     A,C
        CPI     9
        JZ      CH80    ;zase $09
        CPI     0B
```

```
        JZ      CH408   ;tentokrat $08
        STA     PRECH   ;ani jeden z nich
        MVI     A,1     ;vrat to do bufferu
        STA     BUFINT
        CALL    PARAM   ;cti cislo
        MOV     A,L
        STA     PRST    ;nastav pocet r. na stranku
        CALL    BUFE    ;oddelovac pryc
        JMP     STRAN   ;nastav novou stranu
CH80:   CALL    CR      ;udelej CR
        LXI     H,2     ;nastav velikost znaku
        SHLD    SIXX
        SHLD    SIYY
        MVI     A,78    ;a pocet znaku
        STA     PZRA
        RET
CH408:  CALL    CR      ;udelej CR
        LXI     H,3     ;nastav velikost
        SHLD    SIXX
        SHLD    SIYY
        MVI     A,50    ;a pocet znaku
        STA     PZRA
        RET
CH26:   CALL    CR      ;udelej CR
        LXI     H,6     ;nastav velikost znaku
        SHLD    SIXX
        SHLD    SIYY
        MVI     A,28    ;a pocet znaku
        STA     PZRA
        RET
PRINT:  LDA     RDSI    ;podle prepinace znak
        ANI     80
        CZ      PREVOD  ;preved
        MOV     A,C     ;neprevadej
        ANI     80      ;je-li s 8.bitem
        MOV     A,C
        JNZ     PTECK   ;tiskni tecku
        JMP     PRNI    ;neni- tiskni jej
PTECK:  MVI     A,2E    ;kod tecky
        JMP     PRNI    ;vytisknout
NOVR:   PUSH    B       ;schovej si BC
        CALL    CR      ;udelej CR
        CALL    LF      ;jeste LF
        POP     B       ;obnov BC
        RET             ;a zpet
PRNI:   CALL    PCHR    ;jdi tisknout
        LDA     UKZR    ;pocet znaku
        INR     A       ;zvys o jendu
        MOV     B,A
        LDA     PZRA    ;porovnej s maximalnim
        CMP     B
        JZ      NOVR    ;stejne, udelej novou radku
        MOV     A,B
        STA     UKZR    ;uloz to zpet
        RET
PREVOD: LXI     H,TABPR ;adresa prevodni tabulky
        MOV     A,M     ;kod z tab. do A
        CMP     C       ;porovnej s prevadenym
        JZ      VYMEN   ;hura - je to on
        CPI     0       ;neni uz konec tabulky ?
        RZ              ;je - navrat
        INX     H       ;neni - dalsi polozka
        INX     H
        JMP     PREVOD+3;a znovu
VYMEN:  INX     H       ;nasledujici pozice
        MOV     C,M     ;presun
        RET             ;a vrat se

STRAN:  XRA     A       ;znuluj
        STA     UKZR    ;ukazatel znaku
        STA     UKRA    ;ukazatel radek
        LDA     RDSI1   ;testuj typ papiru
        ANI     20
        JZ      STRR    ;-> role
        CALL    MUS2    ;A4 = piskej
        LHLD    STPX    ;vypocti vzdalenost
        MVI     A,1     ;od okraje
        STA     PRZ
        CALL    COMP    ;a nasob -1
        XCHG
        LXI     B,190   ;konstanta Y pro novy zac.
        CALL    MOVR    ;presun
        JMP     NULXY   ;nuluj souradnice
STRR:   CALL    MONOP   ;presun na pozici dalsiho t.
        LXI     D,0
        LXI     B,0FF10 ;konstanta pro vynechani
STRR1:  CALL    MOVR    ;mista
        CALL    CR      ;udelej CR
        CALL    LF      ;jeste LF
        XRA     A       ;znuluj ukazatel radek
        STA     UKRA
        RET             ;navrat
MUS1:   LXI     H,64    ;inicializace zvukoveho
        SHLD    KLOPA   ;generatoru na standartni
        LXI     H,1     ;hodnotu pro 3 typy piskani
        SHLD    KLOPB
        MVI     A,1
        STA     CKLO
        MVI     A,30
        STA     KMIT
        JMP     MUSIC

MUS2:   LXI     H,0FA
        SHLD    KLOPA
        SHLD    KLOPB
        MVI     A,1
        STA     CKLO
        MVI     A,30
        STA     KMIT
        JMP     MUSIC
MUS3:   LXI     H,1F4
        SHLD    KLOPB
        SHLD    KLOPA
        MVI     A,1
        STA     CKLO
        MVI     A,70
        STA     KMIT
        JMP     MUSIC
MUSIC:  LDA     CKLO    ;vlastni generovani zvuku
        STA     CKL01
        LHLD    KLOPA
        SHLD    KLOP1
MUSY:   CALL    BEEP    ;jedno pisknuti
        LHLD    KLOP1
        DCX     H
        MOV     A,L
        ORA     H
        SHLD    KLOP1
        JNZ     MUSY    ;odpiskano ? -> ne
        LDA     CKL01   ;ano
        DCR     A
        STA     CKL01
        JNZ     MUPRO
        LHLD    KLOPB
        SHLD    KLOP2
MUSN:   LDA     KMIT    ;budeme chvili potichu
        CALL    WAITA
        CALL    TLAC    ;testuj tlacitko
        LHLD    KLOP2
        DCX     H
        MOV     A,L
        ORA     H
        JZ      MUSIC
        SHLD    KLOP2
        JMP     MUSN
MUPRO:  MVI     B,68    ;pauza
        CALL    P50
        JMP     MUSIC+6
TLAC:   LDA     RDSI    ;je-li stisk tl.
        ANI     20
        RNZ             ;ne
STISK:  LDA     RDSI    ;ano -> cekej
        ANI     20      ;na jeho pusteni
        JZ      STISK
        POP     H
        LDA     KR025
        ANI     7F
        STA     RDS0    ;a zhasni diodu
        RET

BEEP:   LDA     KLM     ;podle frekvence zmeni
        XRI     20      ;stav reproduktoru
        STA     KLM
        ORI     80      ;+ rozsvicena dioda
        MOV     B,A
        LDA     KR025
        ORA     B
        STA     RDS0
        LDA     KMIT
        CALL    WAITA
        JMP     TLAC    ;a jeste otestuje tlacitko
WAITA:  DCR     A       ;smycka
        RZ
        JMP     WAITA
CR:     XRA     A       ;znuluj
        STA     UKZR    ;pocet znaku
        LXI     H,0
        SHLD    NOPX    ;tisk bude od kraje
        RET
LF:     LDA     UKRA    ;provede LF
        INR     A
        STA     UKRA
        LXI     H,PRST
        CMP     M       ;test na pocet radek
        JZ      STRAN   ;maximalni -> strankuj
        CALL    VYPY    ;vypocti Y
        MVI     A,1
        STA     PRZ
        CALL    COMP
        XCHG
        LHLD    NOPY
        DAD     D
        SHLD    NOPY    ;tisk bude o radku niz
        RET
CHCOL:  LDA     COLR    ;nastavi nasledujici barvu
        INR     A
        CPI     5
        JNZ     COL1
        MVI     A,1     ;byla-li 4 nastavi znovu 1
        JMP     COL1
MONOP:  LHLD    NOPX    ;presun na poz. pristiho znaku
        XCHG
        LHLD    NOPY
```

```
        MOV     B,H
        MOV     C,L
        JMP     MOVA
BASP:   LDA     UKZR    ;provede zpetny krok
        DCR     A
        INR     A
        RZ              ;vrat se jsi-li na zac.radky
        DCR     A
        STA     UKZR
        CALL    VYPX
        MVI     A,1
        STA     PRZ
        CALL    COMP
        XCHG
        LHLD    NOPX
        DAD     D
        SHLD    NOPX    ;X nastav na predchozi znak
        RET
VYPX:   LHLD    SIXX    ;vypocte X podle
        MOV     A,L     ;sirky znaku
        MVI     A,6
        XCHG
        LXI     H,0
        JMP     NASOB
LINUP:  LDA     UKRA    ;provede posun o radek zpet
        DCR     A
        INR     A
        RZ              ;zac. stranky - vrat se
        DCR     A
        STA     UKRA
        CALL    VYPY
        XCHG
        LHLD    NOPY
        DAD     D
        SHLD    NOPY    ;Y nastav o radek zpet
        RET
VYPY:   LHLD    SIYY    ;vypocte Y podle
        MOV     A,L     ;vysky radky
        ANI     0FE
        RRC
        MOV     E,A
        MVI     D,0
        DAD     D
        XCHG
        LXI     H,0
        MVI     A,8
        CALL    NASOB
        RET
PCR:    CALL    CR      ;podle prep. udela CR
        LDA     RDSI1   ;nebo CR + LF
        ANI     80
        RNZ
        JMP     LF
PLF:    CALL    LF      ;podle prep. udela LF
        LDA     RDSI1   ;nebo LF + CR
        ANI     40
        RNZ
        JMP     CR
FAST:   CALL    PARAM   ;nastav krok
        MOV     A,L
        ANI     1
        RLC
        RLC
        RLC
        RLC
        STA     KR025
        ANI     10
        JNZ     F025
        MVI     A,0     ;uprav rychlost
        JMP     SETS+4  ;pro 0.125 mm
F025:   MVI     A,10    ;pro 0.25 mm
        JMP     SETS+4
                        ;prevod SHARP -> ASCII
TABPR:  DB      0D7,10,0FC,17,5E,18
        DB      0F,0C,0E,8,0C,7
        DB      0C6,19,5F,1A,0AE,1B
        DB      0FB,1C,0FF,1E,0CF,1F
        DB      80,5E,93,60,0A1,61
        DB      9A,62,9F,63,9C,64
        DB      92,65,0AA,66,97,67
        DB      98,68,0A6,69,0AF,6A
        DB      0A9,6B,0B8,6C,0B3,6D
        DB      0B0,6E,0B7,6F,9E,70
        DB      0A0,71,9D,72,0A4,73
        DB      96,74,0A5,75,0AB,76
        DB      0A3,77,9B,78,0BD,79
        DB      0A2,7A,0BE,7B,80,7D
        DB      94,7E,0
                        ;tabulka tisku pro test
TSTTB:  DB      2A,2A,2A,20,4D,5A
        DB      2D,38,32,31,20,2A
        DB      2A,2A

MOVA:   CALL    PNUP    ;zvedni pero
        JMP     ABSO    ;presun na X,Y
VECTA:  CALL    PNDW    ;spust pero
        JMP     ABSO    ;presun na X,Y
POINTA: CALL    PNUP    ;zvedni pero
        CALL    ABSO    ;presun na X,Y
        JMP     PNPT    ;udelej bod

MOVR:   CALL    PNUP    ;zvedni pero
        JMP     RELA    ;relativni presun
VECTR:  CALL    PNDW    ;spust pero
        JMP     RELA    ;relativni presun
POINTR: CALL    PNUP    ;zvedni pero
        CALL    RELA    ;relativni presun
        JMP     PNPT    ;udelej bod

PNPT:   CALL    PNDW    ;spust pero
        JMP     PNUP    ;zvedni pero = bod
SPEED:  ADI     8       ;nastav rychlost
        STA     SPED
        RET
TEST:   JMP     VTEST
PORG:   MVI     A,14    ;nastav si rychlost
        STA     SPED
        STA     P185
        PUSH    D       ;schovej zadane X,Y
        PUSH    B
        MVI     A,1
        STA     SPEN    ;pero je dole
        LXI     B,8
        MOV     D,B
        MOV     E,C
        CALL    MOVR    ;par kroku doprava nahoru
        MVI     A,0B    ;uprav rychlost
        STA     SPED
        LXI     H,6A4   ;maximalni X
        SHLD    STPX
        LXI     H,0FE70 ;Y po zalozeni papiru
        SHLD    STPY
        POP     B
        PUSH    B       ;obnov Y - presun na zadane Y
        LXI     D,0     ;okamzite a na 0 do X
        CALL    MOVA
        POP     B       ;obnov Y
        PUSH    B
        LXI     D,50    ;kousek doprava
        CALL    MOVA
        XRA     A
        STA     STPX
        STA     TPLI
        CALL    NULXY   ;znuluj souradnice
        POP     B
        POP     D
        CALL    MOVA    ;presun na zadane X,Y
KTST:   MVI     A,0A
        STA     P185
        LXI     H,3     ;standartni velikost znaku
        SHLD    SIXX
        SHLD    SIYY
        MVI     L,0
        SHLD    SIXY
        SHLD    SIYX
        SHLD    ASCD    ;rezim
        MVI     A,80
        STA     NARW    ;mezera
        STA     PSLI    ;stav pera
        RET
RELA:   LHLD    STPX    ;k X pricti relativni DX
        DAD     D       ;= nova X
        XCHG
        LHLD    STPY    ;totez s Y
        DAD     B
        MOV     B,H
        MOV     C,L     ;a muzes udelat abs. presun
ABSO:   LHLD    STPX    ;DE = zadana Xz
        XCHG            ;HL = skutecna X
        SHLD    STPX    ;DE <=> HL ; X <=> Xz
        SHLD    NOPX    ;a uloz na novou poz.tisku
        CALL    SMR     ;zjisti smer posuvu pro X
        PUSH    H       ;schovej rozdil X a Xz
        PUSH    B       ;schovej Yz
        MOV     C,A     ;C = smer posunu
        POP     D       ;DE = predesle BC t.j. zadana Yz
        LHLD    STPY    ;HL = skutecna Y
        XCHG            ;DE <=> HL ; Y <=> Yz
        SHLD    STPY
        SHLD    NOPY    ;uloz
        CALL    SMR     ;a zjisti smer pro Y
        RLC
        RLC
        RLC
        RLC             ;do vyssich 4 bitu v A
        MOV     B,A
        ORA     C       ;sloz s C
        STA     DIRE    ;<= smer do Y a do X
        POP     D       ;obnov rozdil X a Xz
        RZ              ;jsou-li oba posuny 0 vrat se
        MOV     A,L
        SUB     E
        MOV     A,H
        SBB     D       ;test, ktery posun je vetsi
        MOV     A,C     ;A = smer posunu X
        JC      DALX    ;skoc protoze (Yz-Y)<(Xz-X)
        XCHG            ;DE <=> HL ; (Xz-X) <=> (Yz-Y)
        MOV     A,B     ;A = smer do Y
DALX:   STA     DIR1    ;uloz smer
        SHLD    DIFF    ;a uloz taky rozdil
        LXI     H,14    ;male pozastaveni
```

```
        CALL    FMOT1
        MVI     A,29
        STA     ACCE    ;rozbehova rychlost
        MOV     B,D     ;BC = rozdil
        MOV     C,E
        XRA     A
        SUB     E
        MOV     L,A
        MVI     A,0
        SBB     D
        MOV     H,A     ;HL = DE * -1
        XRA     A
        MOV     A,D
        RAR
        MOV     D,A
        MOV     A,E
        RAR
        MOV     E,A     ;DE = DE / 2
ABS1:   PUSH    H       ;schovej HL a DE
        PUSH    D
        XCHG            ;DE <=> HL
        MOV     H,B
        MOV     L,C     ;HL = BC
        DAD     H
        DCX     H       ;HL = HL * 2 - 1
        DAD     D       ;HL = HL + DE
        LXI     H,ACCE
        JNC     ABS2    ;je-li HL >= 0 skoc
        LDA     P185
        CMP     M
        JZ      ABS2
        DCR     M
ABS2:   MVI     A,28
        SUB     M
        SUB     C
        MVI     A,0
        SBB     B
        JC      ABS3
        INR     M
ABS3:   LHLD    DIFF
        POP     D
        DAD     D
        XCHG
        POP     H
        PUSH    H
        DAD     D
        LDA     DIR1
        JNC     ABS4
        XCHG
        LDA     DIRE
ABS4:   PUSH    D
        CALL    ELEM
        DCX     B
        MOV     A,B
        ORA     C
        POP     D
        POP     H
        JNZ     ABS1
        RET
SMR:    MOV     A,L     ;vrati A = 0 pro skut. = zadane
                        ;       A = 1 pro skut. < zadane
                        ;       A = F pro skut. > zadane
        SUB     E
        MOV     L,A     ;L = L - E
        MOV     A,H
        SBB     D
        MOV     H,A     ;H = H - D - predchozi znam.
        DAD     H       ;HL = HL * 2
        JNC     SMVL    ;-> je-li HL kladne
        XRA     A       ;zaporne - preved na kladne
        SUB     L
        MOV     L,A
        MVI     A,0
        SBB     H
        MOV     H,A     ;HL = -1 * HL
        MVI     A,0F    ;priznak a navrat
        RET
SMVL:   MOV     A,H
        ORA     L
        RZ              ;je-li HL = 0 vrat se
        MVI     A,1
        RET             ;jinak priznak a navrat
WRITE:  XRA     A
        STA     COUNZ
WRIT1:  MOV     A,E
        ORA     A       ;je-li delka nulova
        RZ              ;vrat se
        LDA     COUNZ
        ORA     A       ;neni-li 1. znak
        JNZ     WRIT3   ;skoc
        MOV     A,M     ;je to prvni znak
        ORA     A       ;testuj je-li 0
        JZ      WRIT2   ;je - tak skoc
        MOV     A,M     ;neni, jdi jej vytisknout
        CALL    PCHR
        JMP     WRCON
WRIT2:  INR     A
        STA     COUNZ
        CALL    BS08
WRIT3:  MOV     A,M
        CALL    PCHR4
        MOV     A,M
        ORA     A
        JP      WRCON
        XRA     A
        STA     COUNZ
        CALL    PCHR3
WRCON:  INX     H       ;dalsi pozice znaku
        DCR     E       ;zbyvajici pocet
        JMP     WRIT1   ;a znovu
BS08:   PUSH    H       ;neni-li nastaven bs
        LXI     H,BS    ;na nulu nedela nic
        XRA     A
        CMP     M
        MOV     M,A
        JNZ     BSNA
        LHLD    NOPX    ;jinak nastavi tisk znaku
        SHLD    PREX    ;na novou pozici
        LHLD    NOPY
        SHLD    PREY
BSNA:   POP     H
        RET
PCHR:   PUSH    H
        LXI     H,BS
        CPI     8
        JNZ     NOBS
        MOV     M,A     ;ma-li kod $08
        POP     H       ;pouze jej uloz
        RET             ;a return
NOBS:   ANI     7F      ;pouze 7 bitu
        MOV     C,A
        CALL    BS08    ;vyhodnot byl-li kod $08
        LDA     ASCD
        ORA     A
        JNZ     NODIA   ;skoc pri ASCII rezimu
        MOV     A,C
        CPI     7B
        JC      NODIA   ;skoc neni-li to diakr. znak
        ADI     5       ;zvys na diakr. znamenka
        MOV     C,A
        INR     M
NODIA:  MOV     A,C     ;znak do A
        CPI     10      ;je-li mensi nez 10
        JC      PCHR2   ;skoc
        ANI     0F8     ;ne - znuluj 3 nejnizsi bity
        RAR             ;vydel 2
        RAR             ;      4
        ADI     0FC     ;odecti 3
        ADI     LOW ATB1;+ nizsi byte adresy tab.
        MOV     L,A     ;do L
        MVI     A,0
        ACI     HIGH ATB1
        MOV     H,A     ;do H vyssi byte tabulky
        MOV     A,M
        INX     H
        MOV     H,M
        MOV     L,A     ;do HL adresu generatoru
        MOV     A,C
        ANI     7       ;pouze 3 nejnizsi bity
        MOV     C,A
        JZ      VYZN    ;jsou-li 0 skoc
SKZN:   MOV     A,M     ;nejsou
        ORA     A
        INX     H
        JP      SKZN    ;testuj nejvyssi bit
        DCR     C
        JNZ     SKZN    ;adresu na znak podle nich
VYZN:   MVI     A,8     ;znak v CG vybran
        CALL    PCHR4
PCHR1:  MOV     A,M
        CALL    PCHR4
        MOV     A,M
        INX     H
        ORA     A
        JP      PCHR1
PCHR2:  POP     H
PCHR3:  PUSH    H
        LDA     NARW
        RLC
        RAL
        ADI     0F6
        CMA
        MVI     C,0
        CALL    PCHR5
        POP     H
        RET
PCHR4:  MOV     C,A
        ANI     7
PCHR5:  PUSH    H
        PUSH    D
        PUSH    PSW
        LHLD    SIXX
        XCHG
        LHLD    PREX
        CALL    PCHR8
        XCHG
        MOV     A,C
        ANI     38
        RAR
        RAR
```

```
        RAR
        LXI     H,BS
        ADD     M
        MOV     B,A
        LHLD    SIYX
        XCHG
        CALL    PCHR8
        POP     PSW
        PUSH    H
        LHLD    SIXY
        XCHG
        LHLD    PREY
        CALL    PCHR8
        MOV     A,B
        XCHG
        LHLD    SIYY
        XCHG
        CALL    PCHR8
        PUSH    H
        MOV     A,C
        ANI     40
        JZ      PCHR6
        LHLD    NOPX
        XCHG
        LHLD    NOPY
        MOV     B,H
        MOV     C,L
        LDA     SPEN
        ORA     A
        CZ      MOVA
        POP     B
        POP     D
        CALL    VECTA
        JMP     PCHR7
PCHR6:  CALL    PNUP
        POP     H
        SHLD    NOPY
        POP     H
        SHLD    NOPX
PCHR7:  POP     D
        POP     H
        RET
PCHR8:  INR     E
        DCR     E
        RZ
        PUSH    H
        LXI     H,0
PCHR9:  DCR     A
        JM      PCHR0
        DAD     D
        JMP     PCHR9
PCHR0:  POP     D
        DAD     D
        RET

                        ;proved krok s motory

ELEM:   CALL    DRZ     ;zjisti jsi-li na dorazu
        LHLD    SMOT    ;okamzity stav motoru
        MOV     H,A
        ADD     L       ;pricti krok k motoru
        ANI     0F      ;spodni bity motoru
        MOV     E,A     ;do E
        MOV     A,H     ;druhy motor
        ANI     0F0     ;vyssi bity
        ADD     L       ;pricti krok
        ANI     0F0     ;opet pouze vyssi bity
        ORA     E       ;sloz s E
        STA     SMOT    ;uloz to na stav motoru
        XRA     A
        MOV     H,A
        LDA     ACCE
        RAR
        MOV     L,A     ;HL = rychlost snizena na 1/2
        JMP     NASTM
PNUP:   LDA     SPEN    ;zvedni pero
        DCR     A
        RNZ             ;je-li zvednuto vrat se
        JMP     NASTP
PNDW:   LDA     SPEN    ;spust pero
        DCR     A
        RZ              ;je-li uz dole vrat se
        MVI     A,1
NASTP:  STA     SPEN    ;uloz to na stav pera
        LXI     H,1F4   ;pauza pro pero
NASTM:  PUSH    H
        LHLD    SPEN
        MOV     A,H
        ANI     0E
        ORA     L
        MOV     L,A
        MOV     A,H
        ANI     0E0
        RRC
        ORA     L
        RRC
        CALL    PRELI
        POP     H
FMOT1:  LDA     SPED
FMOT2:  DCR     A
        JNZ     FMOT2
        DCX     H
        MOV     A,H
        ORA     L
        JNZ     FMOT1
        RET

PRELI:  PUSH    B       ;kresli prerusovanou linku
        PUSH    PSW     ;je-li povolena
        ANI     80      ;mas-li pouze presouvat
        JZ      LINR    ;neprerusuj
        LDA     TPLI    ;neprerusuj take
        ORA     A       ;kdyz je typ cary
        JZ      LINR    ;nastaven na plnou
        LDA     AKLI    ;sniz pocet zbyvajicich kroku
        DCR     A       ;s okamzitym stavem pera
        JZ      LINZ    ;je-li 0 musis jej zmenit
        STA     AKLI    ;uloz a neprerusuj
LINR:   POP     PSW     ;vystupni slovo pro motory
        MOV     B,A
        LDA     PSLI    ;nastav pero
        ORI     7F
        ANA     B       ;podle aktualniho stavu
        STA     MOTR    ;posli to na motory
        POP     B
        RET
LINZ:   LDA     PSLI    ;cti stav pera
        ORI     7F      ;ostatni do jednicky
        MOV     B,A
        POP     PSW     ;obnov vystupni slovo
        ANA     B       ;orizni podle pera
        STA     MOTR    ;posli to na motory
        CALL    P50MS   ;pockej na zmenu polohy pera
        LDA     PSLI    ;zinvertuj stav pera
        XRI     80
        STA     PSLI    ;a uloz jej
        LDA     TPLI    ;obnov aktualni
        STA     AKLI    ;typ cary
        POP     B
        RET

P50MS:  MVI     B,19    ;zpozdeni 50 ms
P50:    MVI     C,0F4
P50A:   DCR     C       ;vnitrni smycka
        JNZ     P50A
        DCR     B       ;hlavni smycka
        JNZ     P50
        RET

DRZ:    MOV     L,A
        ANI     0F      ;neni-li posun ve smeru X
        JZ      DRZN    ;doraz netestuj
        CPI     1
        LDA     RDSI
        JZ      DRZR    ;posun doprava, zkus pravu
        ANI     1
        JNZ     DRZN    ;1. doraz neni, zkus druhy
DRZY:   MOV     A,L     ;zjistil jsem doraz
        ANI     0F0     ;zakazuji posun ve smeru X
        PUSH    PSW
        LDA     PPORG   ;je-li PPORG=1
        ORA     A
        JNZ     HAVAR   ;vyhlas havarii
        POP     PSW     ;jinak se vrat do programu
        RET             ;ktery te zavolal
DRZR:   ANI     2       ;jeste druhy doraz
        JZ      DRZY    ;stisknut
DRZN:   MOV     A,L     ;doraz jsem nezjistil
        RET             ;vracim

NULXY:  LXI     H,0     ;okamzita poloha
        SHLD    STPX    ;pera je nastavena
        SHLD    STPY    ;jako zakladni (0,0)
        SHLD    NOPX
        SHLD    NOPY
        RET

;tento pprog. provede celkovy test tiskarny
;vyzaduje papir zalozeny na funkci ORG

VTEST:  LXI     D,186
        LXI     B,0FFB0
        CALL    PORG    ;proved ORG na pozici X,Y
        MVI     A,0A
        STA     SIXX
        STA     SIYY    ;velikost znaku
        ADD     A
        STA     P185
        XRA     A
        STA     NARW    ;mezera
        INR     A
        STA     ASCD    ;ASCII rezim
        MVI     H,10
TCON:   MOV     A,H     ;smycka pro tisk cele tab.
        CALL    PCHR
        INR     H
        MOV     A,H
        ANI     7
        JNZ     TCON    ;dalsi znak
        PUSH    H       ;byla uz radka
        MOV     H,D
        MOV     L,E
        SHLD    NOPX
```

```
        LHLD    NOPY    ;posun na dalsi
        LXI     B,0FF80 ;radek
        DAD     B
        SHLD    NOPY
        POP     H
        MOV     A,H
        ORA     A
        JP      TCON
        JMP     KTST
ATB1:   DW      TB1     ;odkazy do jednotlivych
ATB2:   DW      TB2     ;tabulek generatoru
ATB3:   DW      TB3     ;znaku
ATB4:   DW      TB4
ATB5:   DW      TB5
ATB6:   DW      TB6
ATB7:   DW      TB7
ATB8:   DW      TB8
ATB9:   DW      TB9
ATB10:  DW      TB10
ATB11:  DW      TB11
ATB12:  DW      TB12
ATB13:  DW      TB13
ATB14:  DW      TB14
ATB15:  DW      TB15

;vlastni generator znaku

TB1:    DB      38,7C,74,70,78,80
        DB      8,78,7C,4C,48,32,52
        DB      1B,52,59,80,08,78,7C
        DB      4C,48,12,72,2B,72,69
        DB      80,8,78,7C,4C,48,21
        DB      63,2A,63,5A,80,8,78
        DB      7C,4C,48,21,63,2A,61
        DB      5A,80,8,78,7C,4C,48
        DB      11,71,33,53,23,61,80
        DB      8,78,7C,4C,48,1B,52
        DB      59,69,72,6B,80,3A,4A
        DB      60,24,4A,80
TB2:    DB      0A,7A,60,24,7A,80
        DB      20,64,52,32,64,80,24
        DB      60,52,32,60,80,0,49
        DB      79,7B,74,6B,69,6B,64
        DB      5C,54,4B,4A,51,80,0C
        DB      4B,59,58,48,49,79,7B
        DB      74,2A,68,80,80,8,51
        DB      69,28,6C,2A,4A,4B,54
        DB      80,8,50,54,4C,48,80
TB3:    DB      80,3A,5A,0A,0CA,31,7A,32
        DB      0FB,39,49,0B,7B,2C,68,18
        DB      0DC,10,53,5C,63,61,68,71
        DB      74,3A,0CA,7C,39,70,69,72
        DB      79,1B,52,4B,54,0DB,0C,68
        DB      70,79,7A,73,6B,50,49,4A
        DB      0DC,3B,0F2
TB4:    DB      3B,69,59,0CB,39,6B,5B,0C9
        DB      11,73,24,60,31,0D3,20,64
        DB      32,0D2,12,0C1,20,0E4,0A,0CA
        DB      0FC
TB5:    DB      7C,34,54,4B,49,50,70,79
        DB      7B,0F4,29,7B,0CB,30,79,7B
        DB      74,6C,48,0CC,30,79,7B,74
        DB      6C,63,5C,54,4B,49,0D0,0B
        DB      7B,58,0DC,10,49,4B,54,5C
        DB      63,60,78,0FC,34,7B,79,70
        DB      50,49,4B,54,5C,63,61,0D8
        DB      7C,0F8
TB6:    DB      9,50,58,61,63,6C,74,7B
        DB      79,70,68,61,63,5C,54,4B
        DB      0C9,10,49,4B,54,74,7B,79
        DB      70,68,61,63,0EC,1A,5A,2A
        DB      0EA,1,52,22,0E2,33,61,0D3
        DB      28,6C,18,0DC,31,63,0D1,30
        DB      79,7B,74,6C,5A,52,0A,0CA
TB7:    DB      68,71,73,6C,54,4B,4A,51
        DB      61,6A,63,0CB,7A,4C,21,0E3
        DB      78,7B,74,6C,63,60,63,5C
        DB      54,4B,0C8,34,7B,79,70,50
        DB      49,4B,0D4,78,7B,74,54,4B
        DB      0C8,3C,78,48,4C,20,0E2,78
        DB      7C,20,0E2,34,7B,79,70,50
        DB      49,4B,54,5C,0DA
TB8:    DB      78,3C,4C,20,0E4,9,4B,4A
        DB      7A,79,0FB,10,49,4B,54,0FC
        DB      78,3C,60,0CC,38,48,0CC,78
        DB      62,7C,0CC,78,4C,0FC,9,50
        DB      70,79,7B,74,54,4B,0C9
TB9:    DB      78,7B,74,6C,63,0E0,9,50
        DB      70,79,7B,74,54,4B,49,1A
        DB      0CC,78,7B,74,6C,63,60,0CC
        DB      10,49,4B,54,5C,63,61,68
        DB      70,79,7B,0F4,38,7C,3A,0CA
        DB      38,50,49,4B,54,0FC,38,4A
        DB      0FC,38,49,6A,4B,0FC
TB10:   DB      7C,38,0CC,38,62,7C,22,0CA
        DB      38,7C,48,0CC,38,79,49,0CB
        DB      38,0CC,39,7B,4B,0C9,31,7A
        DB      0F3,0,0C4
TB11:   DB      39,0F2,29,6B,64,54,4B,49
        DB      50,59,5B,54,0CC,78,28,6B
        DB      64,54,4B,0C8,2C,69,60,50
        DB      49,0CC,3C,4C,49,50,60,69
        DB      0EC,18,5C,64,6B,69,60,50
        DB      49,0CC,0A,72,7B,7C,21,0E3
        DB      0,43,4C,6C,69,60,58,51
        DB      0D4
TB12:   DB      78,28,6B,64,0CC,9,4B,4A
        DB      6A,69,3A,0FA,1,42,4B,6B
        DB      6A,3B,0FB,78,2C,58,0CC,9
        DB      4B,4A,7A,0F9,68,60,69,62
        DB      4A,22,6B,64,0CC,68,60,69
        DB      6B,64,0CC,09,50,60,69,6B
        DB      64,54,4B,0C9
TB13:   DB      0,68,6B,64,5C,53,0D0,4
        DB      6C,69,60,58,51,0D4,9,69
        DB      61,6A,6B,0E4,4B,54,5B,59
        DB      60,69,0EC,0C,4B,52,7A,29
        DB      0EB,28,50,49,4B,54,0EC,28
        DB      4A,0EC,28,49,62,4B,0EC
TB14:   DB      6C,28,0CC,0,43,4C,6C,28
        DB      58,51,0D4,28,6C,48,0CC,0B
        DB      52,5A,61,6A,72,0FB,0A,0FA
        DB      9,52,5A,63,6A,72,0F9,20
        DB      69,5B,0E4,39,7B,6B,69,79,80
TB15:   DB      32,0FB,2A,71,7A,73,0EA,39
        DB      72,0FB,31,7A,0F3,39,79,3B
        DB      0FB
        .DEPHASE


        .PHASE  RAM     ;blok promennych

P160:   DS      2
SIXX:   DS      2       ;velikost pisma XX
SIXY:   DS      2       ;velikost pisma XY
SIYX:   DS      2       ;velikost pisma YX
SIYY:   DS      2       ;velikost pisma YY

STPX:   DS      2       ;okamzita poloha pera X
STPY:   DS      2       ;okamzita poloha pera Y
NOPX:   DS      2       ;poloha pristiho tisku zn. X
NOPY:   DS      2       ;poloha pristiho tisku zn. Y

NARW:   DS      1       ;druh mezer mezi znaky
ASCD:   DS      1       ;rezim diakriticky/ASCII

BS:     DS      1
COUNZ:  DS      1       ;citac znaku pri f. WRITE

SPEN:   DS      1       ;okamzity stav pera
SMOT:   DS      1       ;stav krokovych motorku

PREX:   DS      2       ;sour. predchoziho tisku X
PREY:   DS      2       ;-------- " ------------ Y

SPED:   DS      1       ;aktualni rychlost posuvu

ACCE:   DS      1       ;zrychlovani pri rozbehu
DIRE:   DS      1       ;smer do os Y a X
DIR1:   DS      1
DIFF:   DS      2       ;rozdil sour. zadane a skut.
P182:   DS      1
P183:   DS      1
P184:   DS      1
P185:   DS      2

AKLI:   DS      1
PSLI:   DS      1
TPLI:   DS      1

REZI:   DS      1       ;rezim textovy/graficky

COLR:   DS      1       ;aktualni barva

SPD:    DS      1
RTLI:   DS      1

SKUTX:  DS      2       ;poloha 0 od poc. papiru v X

PPORG:  DS      1       ;druh funkce dorazu
VOLB:   DS      1
                        ;promenne pro rizeni zvuk.
                        ;       generatoru
KMIT:   DS      1       ;kmitocet piskani
KLOP1:  DS      2       ;       pomocne
CKLO1:  DS      1       ;       promenne
KLOP2:  DS      2       ;       pro generator
KLOPA:  DS      2       ;delka pauzy mezi pisk.
KLOPB:  DS      2       ;delka pauzy mezi cykly
CKLO:   DS      1       ;pocet pisknuti
KLM:    DS      1       ;stav reproduktorku

SIZE:   DS      1       ;velikost znaku pro GPRINT

UKRA:   DS      1       ;ukazatel radek
UKZR:   DS      1       ;ukazatel znaku

STWP:   DS      1
PUYE:   DS      1
PUXX:   DS      2
```

```
PRST:   DS      1       ;pocet radek na stranku
PZRA:   DS      1       ;pocet znaku na radek

PLNB:   DS      1
POMB:   DS      1

PITCH:  DS      2       ;roztec dilku na ose
REPE:   DS      1       ;pocet dilku na ose

PRECH:  DS      1       ;cist z bufferu nebo z BUFINT
BUFINT: DS      1       ;pomocne prodlouzeni bufferu

KR025:  DS      1       ;krok 0.125 nebo 0.25
PRZ:    DS      1

CK:     DB      0       ;pomocne promenne pro rizeni
CW:     DB      0       ;ukazovatka bufferu
ZW:     DB      0
BUFF:   DS      0FF     ;vlastni buffer 1/4 kbyte
        DS      1
CGR:    DS      1FF
        DS      30      ;prostor pro stack
STACK:
        END     0
```

Manuál pro plotter – tiskárnu

Formát příkazů

Je 5 typů formátu příkazů, jak je popsáno dále.

1. Příkaz skládající se pouze z jednoho znaku (bez další specifikace) (A, H, I).
2. Příkaz, skládající se ze znaku a jednoho dalšího parametru (L, C, S, Q).
3. Příkaz, skládající se ze znaku a dvou parametrů. "," se používá k oddělení parametrů a kód CR na zakončení souboru parametrů.
4. Příkaz plus znakový řetězec (P).
5. Příkaz skládající se ze znaku a tří dalších parametrů (X).

"," se používá k oddělení parametrů.

Zadání parametrů

1. Mezery, předřazené parametru jsou ignorovány.
2. Každé číslo, kterému je předřazeno znaménko "-" je považováno za záporné.
3. Každý parametr je zakončen "," nebo kódem CR. Jestliže jako specifikaci parametru zadáme jiný znak, než číslo, všechny znaky následující tento znak jsou ignorovány.

Zkrácené formáty zadání

1. Libovolný jednoznakový příkaz může být následován příkazem bez toho, abychom jej museli oddělovat kódem CR. Např. „HD100,200" CR má stejnou hodnotu jako „H" CR „D100,200" CR.
2. Libovolný dvouznakový příkaz (znak + parametr) může být následován libovolným příkazem, musí však být oddělený ",". Např. „L0,S1,Q0,C1D100,200" CR je platný výraz.

Změny hodnot vnitřních proměnných při změně režimu

Při přechodu z grafického do textového režimu dochází ke změnám hodnot vnitřních proměnných:

> *Souřadnice X a Y.*
Y-ová souřadnice je nastavena na 0 a začátek je přiřazen levé straně plochy, na kterou je možno psát.

> *Směr tisku znaku.*
Parametr Q je nastaven na 0.

> *Velikost znaku*
stupeň velikosti znaku je nastaven na 1 (80 zn. na řádek).

> *Typ čáry.*
Druh čáry je nastaven na 0 (plná čára).

Velikost znaku

> Po zapnutí tiskárny je velikost znaku automaticky nastavena na stupeň 1 (80 znaků na řádek). Stupeň velikosti je možno později měnit kontrolními kódy a příkazy.

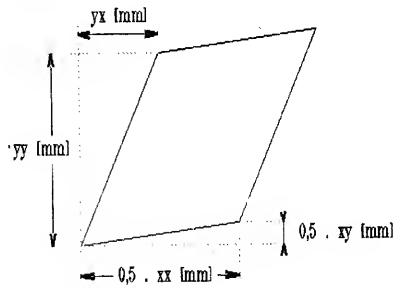
> V grafickém režimu je možno měnit velikost písma v rozsahu 0 až 63.

> Při přepnutí z grafického do textového režimu je nastaven stupeň velikosti znaku na 1.

Příkazy grafického režimu

Počítač řídí tiskárnu v grafickém režimu následujícími příkazy. Slova uvedená v závorkách, jsou příkazy v jazyce SHARP BASIC, které mají stejnou funkci.

Název příkazu	*Formát a funkce*
Druh čáry	**Lp** (p=0 až 15) Určuje druh čáry (plná nebo tečkovaná) a rozteč teček. p=0 platí pro plnou čáru, p=1 až p=15 pro různé tečkované čáry.
Základní poloha (PHOME)	**H** Zdvihne pero a vrátí jej do základní polohy.
Inicializuj základní polohu (HSET)	**I** Nastaví současnou polohu pera jako základní (x=0 a y=0).
Kreslí (LINE)	**Dx,y** (–9999<=x,y<=9999) Kreslí čáru ze současné polohy pera do bodu x, y.
Kreslí (RLINE)	**Jx,y** (–9999<=x,y<=9999) To samé jako předchozí příkaz, ale kreslí do bodu určeného relativními souřadnicemi x, y.
Přesun (MOVE)	**Mx,y** (–9999<=x,y<=9999) Zdvihne pero a přesune jej do bodu určeného souřadnicemi x, y.
Přesun (RMQVE)	**Rx,y** (–9999<=x,y<=9999) To samé jako předchozí příkaz, ale do bodu určeného relativními souřadnicemi x, y.
Bod	**Yx,y** (–9999<=x,y<=9999) Zdvihne pero a přemístí jej do bodu x,y a nakreslí bod.
Bod	**Wx,y** (–9999<=x,y<=9999) Totéž jako předchozí povel, ale bod udělá na místě určeném relativními souřadnicemi x, y.
Změna barvy (PCOLOR)	**Cn** (n=0 až n=3) Změní barvu pera podle čísla n. n=0 . . . černá n=1 . . . modrá n=2 . . . zelená n=3 . . . červená
Nastav stupeň	**Sn** (n=0 až 63) Určuje stupeň velikosti znaku.
Otoč znak	**Qn** (n=0 až 3) Určuje směr kterým se píší znaky.
Tiskni	**Pc1c2c3c4 . . . cn** Tiskne znaky c1, c2 až cn
Osa	**Xp,q,r** (p=0 nebo 1; q=–999 až 999, r=1 až 255) Je-li p=1 kreslí X-ovou osu, je-li p=0 kreslí osu Y-ovou, q určuje rozteč značek na ose a r jejich počet.
Nastav velikost	**Zxx,xy,yx,yy** Obdoba příkazu Sn + Qn, ale poskytuje značně větší možnosti. Nastaví směr, sklon a velikost písma.



XX . . . . . šířka písma v 1/2 | mm |,
XY . . . . . odklon od x-ové osy v 1/2 | mm |,
YX . . . . . odklon od y-ové osy v | mm |,
YY . . . . . výška písma v | mm |.

Několik příkladů volby tisku příkazem „Z":

XX,XY,YX,YY	tvar	výška	šířka	řádka
1,0,0,1	kolmý	1 mm	0,5 mm	vodorovně
4,0,0,7	štíhlý kolmý	7 mm	2 mm	vodorovně
4,0,1,4	skloněn vpravo	4 mm	2 mm	vodorovně
0,6,–4,0	širší, kolmý	4 mm	3 mm	svisle vzhůru
4,4,–4,4	kolmý	5,7 mm	2,3 mm	šikmo vzhůru

Nastav rychlost	**Tn** (n=–10 až 30) Nastaví rychlost tisku, n=–10 je největší a n=30 nejmenší rychlost.
Nastav krok	**Fn** (n=0 nebo 1) Nastaví krok 0.125 mm pro

n=0 nebo 0.25 mm pro n=1. Při kroku 0.125 mm se nastaví stupeň rychlosti na 0 a při kroku 0.25 mm je nastavován stupeň 16, což je třeba vzít v úvahu při nastavování rychlosti příkazem „T".

Nastav diakritický režim **Bn** (n=0 nebo 1)

Nastaví diakritický režim při n=0. Je-li n=1 nastaví režim ASCII. Při tisku diakritických znamének nedochází k přesunu na tisk následujícího znaku, ale další znak se píše pod znaménko.

Tabulka odpovídajících znaků:

ASCII režim	*diakr. režim*
{	5Bh čárka
|	5Ch kroužek
}	5Dh háček

Nadefinuj vlastní znak **En,c1c2c3 . . c32** (n=1 až 15)

Za znakem „E" následuje číslo znaku n z rozsahu 1 až 15 a poté následují kódy podle tabulky uvedené v příloze tohoto manuálu. Může jich být maximálně 32, je-li jich méně, musí končit nastaveným 8.bitem.

Pro definici takového znaku je nejprve nutné nakreslit základní tvar (složený z úseček) do rastru 5×8 bodů, jednotlivým vrcholům (tj. koncům úseček) přiřadit čísla A1,A2, . . . ,AN **Tab. 2.**

Horní čísla znamenají posun s písátkem nahoře, dolní posun s písátkem dolů. Poslední člen řetězce, který zajistí zvednutí písátka a návaznost pro psaní dalších znaků, musí mít 8.bit nastaven na 1 (tj. 80H). Dále lze ještě ukončovací člen (80H sloučit s posledním vrcholem kreslené úsečky (tj. (An+80H)).

Tabulka 2.

dec	hex										
56	38	57	39	58	3A	59	3B	60	3C		
120	78	121	79	122	7A	123	7B	124	7C		
48	30	49	31	50	32	51	33	52	34		
112	70	113	71	114	72	115	73	116	74		
40	28	41	29	42	2A	43	2B	44	2C		
104	68	105	69	106	6A	107	6B	108	6C		
32	20	33	21	34	22	35	23	36	24		
96	60	97	61	98	62	99	63	100	64		
24	18	25	19	26	1A	27	1B	28	1C		
88	58	89	59	90	5A	91	5B	92	5C		
16	10	17	11	18	12	19	13	20	14		
80	50	81	51	82	52	83	53	84	54		
8	8	9	9	10	0A	11	0B	12	0C		
72	48	73	49	74	4A	75	4B	76	4C		
0	0	1	1	2	2	3	3	4	4		
64	40	65	41	66	42	67	43	68	44		

Příklad vytvoření znaku „A":

```
PRINT/P "E1,"; CHR$($08,$7A,$4C,$21,$E3)
```

Vytiskni nadefinovaný znak **Gn** (n=1 až 15)

Vytiskne dříve nadefinovaný znak čísla n z rozsahu 1 až 15.

Kontrolní kódy používané v textovém režimu

Všechny kódy platí při přepínači ASCII/MZ800 nastaveném do polohy ASCII.

Textový kód ($01).
Přepíná tiskárnu do textového režimu.

Grafický kód ($02). (Jako příkaz PMODE v jazyku Basic)
Přepíná tiskárnu do grafického režimu.

Řádek zpět ($03). (Jako příkaz PSKIP v jazyku Basic)
Posune papír o 1 řádku zpět. Obsah čítače se přitom zmenší o 1. Stojí-li pero na začátku nové stránky, neprovádí nic.

Zkouška per ($04). (Jako prikaz PTEST v jazyku Basic)
Nejprve vytiskne následující text, potom nastaví rozměr tisku 1 (80 znaků na řádek).

* * * MZ-821 * * *

Tisk předdefinovaného znaku ($05)+(ASCII)2+(ASCII)1+($0D).
Vytiskne předefinovaný znak čísla složeného z (ASCII)2 a (ASCII)1 z rozsahu 1 až 15.

Zmenšený tisk ($09)+($09)+($09).
Zmenšuje tisk ze stupně 1 na stupeň 0 (120 zn. na řádek).

Zrušení zmenšeného tisku ($09)+($09)+($08).
Zvětšuje tisk ze stupně 0 na stupeň 1 (80 zn. na řádek).

Nastavení čítače řádek. (Jako příkaz PAGE v jazyku BASIC).
($09)+($09)+(ASCII)2+(ASCII)1+(ASCII)0+($0D)
Určuje počet řádek na stránku tříbajtovým kontrolním kódem ASCII. Největší počet je 255. Po zapnutí nebo resetování systému je čítač řádek nastaven na hodnotu 66.

Řádek vpřed ($0A). (Jako příkaz PSKIP v jazyku BASIC).
Posune papír o jeden řádek vpřed. Obsah čítače řádků se zvětší o 1.

Zvětšený tisk ($0B).
Zvětšuje tisk ze stupně 1 na stupeň 2 (40 zn. za řádek).

Zrušení zvětšeného tisku ($07).
Zmenšuje tisk ze stupně 2 na stupeň 1 (80 zn. na řádek).

Návrat vozíku ($0D).
Přesune vozík (písátko) do levé krajní polohy.

Krok zpět ($08).
Přesune vozík o jednu pozici zpět. Je-li v levé krajní poloze, kód nemá žádnou funkci.

Nová strana ($0C).
Příkaz přesune papír na začátek další strany a zároveň vynuluje čítač řádek.

Následující barva ($1D).
Změní barvu pera na následující podle tabulky uvedené výše.

Při výše uvedeném přepínači nastaveném do polohy MZ800 platí kódy podle následující tabulky:

Nová strana	$0F
Zpětný krok	$0E
Zrušení zvětšeného tisku	$0C

Automatický test

Tiskárna umí 2 druhy testu. První se spouští řídicím kódem ($04), nebo tlačítkem TEST na ovládacím panelu. Druhy spustíme stiskem tlačítka TEST a současným zapnutím tiskárny.

Ovládací prvky na čelní stěně tiskárny

Tlačítko STOP –slouží k pozastavení práce plotteru,tiskárna vydává tón a čeká na stisk tlačítka READY.

Tlačítko TEST –provede to samé jako řídící kód $04, tedy test tiskárny.

Tlačítko READY –při jakémkoli zvukovém signálu potvrzuje tímto tlačítkem uživatel prováděnou operaci.

Přepínač PEN/NOT –slouží k invertování okamžitého stavu pera, provádí se technicky nezávisle na činnosti plotteru.

Přepínač ASCII/MZ800 –určuje mají-li se znaky přepočítávat (režim MZ800) či ne.

Přepínač COLOR/NO –při příchodu instrukce pro změnu barvy a nastaveném přepínači do polohy COLOR vydává pískání odpovídající požadované barvě a čeká na stisk tlačítka READY.

Dioda RDY –svitem určuje čekání na stisk tlačítka READY.

Dioda PWR –signalizuje zapnutí plotteru.

Dioda BUF –signalizuje vyčítání z bufferu, nebo svítí-li trvale, určuje prázdný buffer.

Systémové přepínače na zadní stěně

Přepínač CR/CR+LF –v poloze CR+LF při příchodu kódu CR provede i kód LF.

Přepínač LF/LF+CR –v poloze LF+CR při příchodu kódu LF provede i kód CR.

Přepínač A4/ROLE –určuje formát papíru.

Přepínač RDP/RDP –určuje polaritu signálu RDP (hardwerově).

Přepínač RDA/RDA –určuje polaritu signálu RDA (hardwerově).

EMULÁTOR TERMINÁLŮ CM7202/CM7209 EMU89

Ing. Petr Kandera, *Krasnoarmějců 2, 704 00 Ostrava 3*

EMU89 je program pro emulaci terminálů CM7202 a CM7209 na počítači typu IBM PC pod operačním systémem MS-DOS. IBM PC se připojuje jako terminál po sériové lince (RS232C, proudová smyčka).

Po spuštění EMU89 se zobrazí hlavní menu:

1. Emulace terminálu CM7202.
2. Emulace terminálu CM7209.
3. Změna parametrů přenosu.

Volit z těchto možností lze pomocí šipek a klávesy ENTER nebo přímo číslem volby. Kromě těchto možností je možné zobrazit popis funkčních kláves pro oba typy emulovaných terminálů stiskem klávesy K. Pokud do asi 5 sekund nestiskneme žádnou klávesu, automaticky se nastartuje „vysvícená" volba. (Pozn.: časový údaj platí pro CPU 8088 a takt 9,54 MHz).

Pro svou činnost vyžaduje EMU89 v adresáři, ze kterého byl spuštěn, konfigurační soubor EMU89.CFG, ve kterém má uloženy parametry přenosu (číslo sériového portu, přenosovou rychlost, počet datových bitů, počet stop bitů, typ parity), číslo volby, která se po spuštění automaticky aktivuje, a označení diskové jednotky, na které bude hledán systémový soubor COMMAND.COM při práci s operačním systémem v emulátoru.

Všechny parametry, s výjimkou posledního, má uživatel možnost měnit volbou z pull-down menu. Poslední parametr se při instalaci vždy nastaví na C (harddisk). Konfigurační soubor lze měnit také libovolným textovým editorem.

Pokud v okamžiku startu neexistoval konfigurační soubor EMU89.CFG, automaticky bude nastartována volba 3 (Změna parametrů přenosu) a hlavní menu bude zobrazeno až po vytvoření konfiguračního souboru.

Program se ukončuje z hlavního menu pomocí CTRL-A.

Změna parametrů přenosu

Parametry se mění výběrem z menu pomocí šipek a klávesy Enter, jak je vysvětleno v dolní části obrazovky. Klávesa Esc způsobuje vždy návrat o krok zpět beze změny hodnoty. Změnu parametrů lze ukončit stiskem Esc, pak se změny neuloží a platí pouze do ukončení programu, nebo stiskem CTRL-A – změny budou navíc uloženy do souboru. Pokud však před spuštěním neexistoval konfigurační soubor, i ukončení změn parametrů stiskem Esc způsobí uložení parametrů.

Emulace

Při emulaci se chová PC jako zvolený typ terminálu s těmito funkčními klávesami:

F3 Lokální výmaz obrazovky.
F4 Popis funkčních kláves (Help). Tato funkce odpovídá stisku klávesy K v hlavním menu.
F9 Změna parametrů přenosu. Tato funkce je ekvivalentní volbě 3 v hlavním menu.
F10 Odskok do prostředí MS-DOS. Návrat zpět je možný zadáním sekvence EXIT. Pokud nebude nalezen COMMAND.COM na zvoleném disku, dojde po chybovém hlášení k návratu do emulace.
CTRL-A Ukončení emulace a návrat do hlavního menu.
Enter CR – „Návrat vozu" (znak 13D).
End LF – skok na nový řádek (znak 10D).
Delete Delete – výmaz znaku.
Back space Back space (znak 08D).

Pro emulátor CM 7202 dále platí:
F1 EOL – Výmaz řádku od kurzoru doprava.
F2 EOS – Výmaz obrazovky od kurzoru dolů.
Home Home – Přesun kurzoru do levého horního rohu obrazovky.

*(*Pozn.*: tyto funkční klávesy mají při emulaci CM7209 stejný význam jako neoznačené tři klávesy terminálu.)

Zbývající klávesy PC mají stejný význam jako klávesy terminálů nebo nemají žádnou funkci.

Nároky na technické vybavení

Program EMU89 je určen pro počítače standardu IBM PC, pracující pod operačním systémem MS-DOS. Program byl zkoušen na počítačích Commodore PC20-III, PP06, Leonard AT, IBM PC-XT a IBM PS/2-30 s operačními systémy MS-DOS V3.20, V3.21, V3.30 a PPDOS V3.0. Vyžaduje grafickou kartu CGA nebo EGA (pro kartu Hercules stačí upravit atributy Text color z Text Background); v .EXE formě je dlouhý 22784 bajtů a v paměti zabírá asi 45 kB. Počítač musí být vybaven standardním sériovým portem COM1 nebo COM2.

Stručný popis programu

Program je napsán v jazyce TURBO-PASCAL V4.0 (délka zdrojového programu je 520 řádků).

Program EMU89 se skládá ze čtyř základních částí:

1. pomocné globální procedury a funkce,
2. vlastní emulace,
3. procedury a funkce pro změnu parametrů,
4. hlavní program.

Hlavní program zajišťuje tvorbu hlavního menu a provádění základních operací s ním. Používá mimo jiné proceduru Keys pro zobrazení popisu funkčních kláves (Help).

Změnu parametrů výběrem z pull-down menu a vytváření konfiguračního souboru má na starosti procedura

Init,

hlavní procedura změny parametrů. Provádí výpis nastavených hodnot, volbu parametrů atd. Využívá procedury: Default (zobrazení nastavených hodnot), Save (ukládání parametrů do souboru), Mainwindow (okno pro změnu parametrů), Pspdf a Speed (volby jednotlivých parametrů). Dále používá procedury společné i pro hlavní program: Bottom (nápověda) a Arrows (pohyb kurzoru).

Emulace využívá tří hlavních procedur:

Emulace – Nastavení parametrů a volba typu emulátoru,
Emu1 – emulace CM7202,
Emu2 – emulace CM7209

a další pomocné procedury a funkce. Zjednodušeně lze popsat její činnost následovně:

Při spuštění emulace se nastaví parametry sériového portu a vektor přerušení. Přerušení při příjmu znaku na sériový port je obsluhováno procedurou Readcom, která ukládá znaky do pole, ze kterého jsou po zpracování znaky zobrazeny na obrazovce. Mezitím se soustavně testuje klávesnice – běžné znaky jsou vysílány přímo na port, funkční klávesy provádějí popisné funkce. Na obrazovce se zobrazují pouze znaky přicházející na sériový port.

Pomocné globální procedury a funkce zajišťují mimo jiné výstup znaku na port (OutChar), úschovu a obnovení obrazovky (SaveScreen, LoadScreen), odskok do MS-DOSu (Branch), inicializaci portu (Initcom), povolení a zablokování přerušení (SioInt On, SioInt Off), výmaz obrazovky od kurzoru dolů a doprava (ClrEos), přečtení znaku z pole s informací, zda se jedná o nový znak (Inbyte), zobrazení popisu funkčních kláves, změnu parametrů přenosu (Initid) apod.

Program lze snadno přizpůsobit individuálním požadavkům a vytvořit v něm vlastní speciální funkce, např. hardcopy obrazovky, nebo přiřadit dalším funkčním klávesám nové funkce.

Práce s emulátorem je jednoduchá a pohodlná, zvláště když uvážíme, že při práci s ním lze využít celou škálu programů přístupných pod MS-DOSem – v emulátoru lze provést odskok do MS-DOSu, pak v něm libovolně dlouho pracovat a kdykoliv se vrátit zpět do emulátoru.

```
{Emulator terminalu CM7202/CM7209}
{            na IBM PC            }
{            Verze 3.4            }
{       (c) 1989 - PKSoft         }


program EMU89;
{$K-U-C-}
{$M 20000,0,4000}
uses Crt,Dos;
label 5;
const
  sp=' ';comu1=$3F8;comu2=$2F8;
  EOI:byte=$20;BuffTop=10000;
  ter1='C M 7 2 0 2';ter2='C M 7 2 0 9';
  IntContr:integer=$20;q=#219;u=#220;
  IRQ_Mask:integer=$21;h=#223;
type
  strarr=array[1..8] of string[30];
var
  f:text;j:string[30];newbyte:boolean;
  term:string[12];r,z,v:string[2];
  keyout,c,drv:char;ScrSeg:word;
  p,x,d,b,s,t,m,e:strarr;
  ByteIn,ComInt,xx,yy,zac:byte;
  altcom,exitsave,screen:pointer;
  Inbuff:array[0..BuffTop] of byte;
  com,comnr,bd,db,sb,par,yl,yk:word;
  Origmode,Inpoint,Outpoint,i:word;

procedure ReadCom;
  Interrupt;
begin inc(Inpoint);
  if(Inpoint>Bufftop) then Inpoint:=0;
  Inbuff[Inpoint]:=port[com];
  port[IntContr]:=EOI;
end;

function Inbyte(var pb:boolean):byte;
begin pb:=(Outpoint<>Inpoint);
  if pb then begin inc(Outpoint);
   if Outpoint>BuffTop then Outpoint:=0;
     InByte:=InBuff[Outpoint];end;
 end;

procedure OutChar(z:char);
begin while(port[com+5] and $20 =0) do;
  port[com]:=byte(z);
end;

procedure SioInt_off;
begin
Port[IRQ_Mask]:=(Port[IRQ_Mask] or $18);
end;

procedure SioInt_on;
begin
Port[IRQ_Mask]:=
               (Port[IRQ_Mask] and $E7);
end;

procedure Initcom(comport:integer;
        baud:real;data,stop,parit:byte);
var divisor:integer;local:byte;
begin divisor:=round(115200.0/baud);
  port[comport+3]:=$80;
  port[comport]:=  lo(divisor);
  port[comport+1]:=hi(divisor);
  local:=0;case data of 8:local:=3;
    7:local:=2;6:local:=1;end;
  if stop=2 then local:=local or 4;
  case parit of 2:local:=local or $18;
    1:local:=local or $08;end;
port[comport+3]:=local;
port[comport+1]:=1;port[comport+4]:=$0B;
local:=port[comport];
end;

procedure ClrEos;
var i:byte;
begin ClrEol;xx:=wherex;yy:=wherey;
  for i:=yy+1 to 24 do begin
      gotoxy(1,i);ClrEol;end;
  gotoxy(xx,yy);
end;

procedure Title;
begin Textbackground(0);
window(1,1,80,25);clrscr;
Textcolor(0);Textbackground(7);
Insline;gotoxy(16,1);writeln
('E M U L A C E   T E R M I N A L U   ',
term);Textcolor(14);Textbackground(0);
window(1,2,80,25);
end;

{$F+}procedure Myexit {F-};
begin Textmode(Origmode);SioInt_off;
  ByteIn:=port[com];ExitProc:=Exitsave;
  setintvec(comInt,altCom);
end;

procedure SaveScreen;
begin if (Lo(LastMode)=7) then
       ScrSeg:=$B000 else ScrSeg:=$B800;
   Move(Ptr(ScrSeg,0)^,Screen^,4000);
end;

procedure LoadScreen;
begin if (Lo(LastMode)=7) then
       ScrSeg:=$B000 else ScrSeg:=$B800;
   Move(Screen^,Ptr(ScrSeg,0)^,4000);
end;

procedure Init(u:byte);forward;
procedure Keys;forward;

procedure Initia(u:byte);
begin GetMem(Screen,4000);xx:=wherex;
   yy:=wherey;SaveScreen;
   if u=0 then Init(1)
          else Keys;LoadScreen;
   FreeMem(Screen,4000);Textcolor(14);
   Textbackground(0);window(1,2,80,25);
   gotoxy(xx,yy);
end;

procedure Branch;
begin GetMem(Screen,4000);xx:=wherex;
 yy:=wherey;SaveScreen;
 Textbackground(blue);window(1,1,80,25);
 clrscr;writeln('Navrat do emulatoru',
                ' -> <EXIT>');
 j:=drv+':\COMMAND.COM';Exec(j,'');
 if DosError<>0 then begin writeln(#10,
 'Nenalezen soubor COMMAND.COM');
 write(#7);Delay(2000);end;
 Initcom(com,bd,db,sb,par);
 SetIntvec(ComInt,@readcom);
 SioInt_on;LoadScreen;
 FreeMem(Screen,4000);Textcolor(14);
 Textbackground(0);window(1,2,80,25);
 gotoxy(xx,yy);
end;


procedure Emu1;
label 10;
begin while (keyout<>#1) do begin
     ByteIn:=InByte(newbyte);
     ByteIn:=(ByteIn and $7F);
     if newbyte then begin
        if i=2 then begin
           xx:=ByteIn-31;gotoxy(xx,yy);
           i:=0;goto 10;end;
        if i=1 then begin
```

```
          yy:=ByteIn-31;Inc(i);
          goto 10;end;
case ByteIn of 0:;7:write(#7);
 8:gotoxy(wherex-1,wherey);
 9:gotoxy(wherex+
         (8-(wherex mod 8)),wherey);
11:gotoxy(wherex,wherey+1);
24:gotoxy(wherex+1,wherey);
26:gotoxy(wherex,wherey-1);
27:Inc(i);29:gotoxy(1,1);30:ClrEol;
31:ClrEos;else write(char(ByteIn));
  end;end;
10:if keypressed then begin
keyout:=readkey;
if not (keyout=#1) then begin
if keyout=#0 then begin keyout:=readkey;
case ord(keyout) of 83: OutChar(#127);
79: OutChar(#10);72: OutChar(#26);
80: OutChar(#11);75: OutChar(#8) ;
77: OutChar(#24);71: OutChar(#29);
59: OutChar(#30);60: OutChar(#31);
61: clrscr;62: Initia(1);67: Initia(0);
68: Branch;15..25,30..38,46..50,63..66,
84..140,3,44,73,81,82:;end;end
else OutChar(keyout);end;end;end;
end;


procedure Emu2;
label 10;
begin while (keyout<>#1) do begin
     ByteIn:=InByte(newbyte);
     ByteIn:=(ByteIn and $7F);
     if newbyte then begin
       if i=3 then begin
          xx:=ByteIn-31;gotoxy(xx,yy);
          i:=0;goto 10;end;
       if i=2 then begin
          yy:=ByteIn-31;Inc(i);
          goto 10;end;
       if i=1 then begin Dec(i);
          case ByteIn of
           65:gotoxy(wherex,wherey-1);
           66:gotoxy(wherex,wherey+1);
           67:gotoxy(wherex+1,wherey);
           68:gotoxy(wherex-1,wherey);
   72:gotoxy(1,1);75:ClrEol;74:ClrEos;
           89:i:=i+2;else;end;
           goto 10;end;
        case ByteIn of
0:;7:write(#7);9:gotoxy(wherex+
            (8-(wherex mod 8)),wherey);
27:Inc(i);else write(char(ByteIn));
   end;end;
10: if keypressed then begin
          keyout:=readkey;
          if not (keyout=#1) then begin
if keyout=#0 then begin keyout:=readkey;
case ord(keyout) of
83: OutChar(#127);79: OutChar(#10);
72: begin OutChar(#27);OutChar(#65);end;
80: begin OutChar(#27);OutChar(#66);end;
75: begin OutChar(#27);OutChar(#68);end;
77: begin OutChar(#27);OutChar(#67);end;
71: begin OutChar(#27);OutChar(#82);end;
59: begin OutChar(#27);OutChar(#81);end;
60: begin OutChar(#27);OutChar(#80);end;
61: clrscr;62: Initia(1);67: Initia(0);
68: Branch;15..25,30..38,46..50,63..66,
84..140,3,44,73,81,82:;end;end
    else OutChar(keyout);end;end;end;
end;

procedure Emulace;
begin Title;i:=0; SioInt_off;
if comnr=1 then begin
          com:=comu1;ComInt:=$0C;end
else begin com:=comu2;Comint:=$0B;end;
   Initcom(com,bd,db,sb,par);
   SetIntvec(ComInt,@readcom);
   Inpoint:=0;Outpoint:=0;
   SioInt_on;keyout:=#$FF;OutChar(#13);
   if term=ter1 then Emu1 else Emu2;
end;

procedure Arrows(ms:strarr;
                 ds,l,o:integer);
procedure Move(i:integer);
begin write(ms[wherey]);
gotoxy(wherex-ds,wherey+i);
if l=0 then Textbackground(0)
       else Textbackground(green);
  write(ms[wherey]);
  gotoxy(wherex-ds,wherey);
if l=0 then Textbackground(white)
       else Textbackground(0);
end;
begin c:=readkey;case ord(c) of
72: if wherey=1   then Move(o)
                  else Move(-1);
80: if wherey=o+1 then Move(-o)
                  else Move(1);
     else write(#7);end;
end;

procedure Keys;
begin Textbackground(white);
window(1,1,80,25);clrscr;Textcolor(14);
gotoxy(7,24);write('Zbyvajici klavesy',
' PC odpovidaji terminalu nebo nemaji',
' zadny vyznam');gotoxy(30,22);
write('Navrat zpet -> <Esc>');
gotoxy(24,2);write('POPIS FUNKCNICH ',
'KLAVES EMULATORU');Textbackground(0);
window(10,4,35,20);clrscr;
Textbackground(green);
   write(' T E R M I N A L   CM7202 ');
   write('                         ');
   write(' klavesa           funkce ');
   gotoxy(1,5);Textbackground(0);
   write(' F1                   EOL ');
   write(' F2                   EOS ');
   write(' F3       Vymaz obrazovky ');
   write(' F4          Popis klaves ');
   write(' F9       Zmena parametru ');
   write(' F10       Odskok do DOSu ');
   write(' Enter                CR ');
   write(' End                  LF ');
   write(' Home               HOME ');
   write(' Delete(Del)         DEL ');
   write(' Backspace            BS ');
   write(' Ctrl-A  Ukonceni emulace ');
   Textbackground(0);window(45,4,70,20);
   clrscr;Textbackground(green);
   write(' T E R M I N A L   CM7209 ');
   write('                         ');
   write(' klavesa           funkce ');
   gotoxy(1,8);TextBackground(0);
   write(' Enter                CR ');
   write(' End                  LF ');
   write(' Delete(Del)         DEL ');
   write(' Backspace            BS ');
   write(' F3       Vymaz obrazovky ');
   write(' F4          Popis klaves ');
   write(' F9       Zmena parametru ');
   write(' F10       Odskok do DOSu ');
   write(' Ctrl-A  Ukonceni emulace ');
   window(1,1,80,25);gotoxy(47,22);
   repeat repeat until keypressed;
    c:=readkey;if c<>#27 then write(#7);
   until c=#27;
end;


procedure Bottom(u:byte);
begin Textbackground(0);Textcolor(14);
```

```
window(4,22,77,24);clrscr;gotoxy(2,1);
if u=0 then gotoxy(19,1);
write('    Volba moznosti :     ');
Textcolor(10);
write(#24,#25,' +   Enter');
Textcolor(14);if u=1 then begin
write('   nebo   primo ');Textcolor(10);
write(' cislo ');Textcolor(14);
write(' volby');gotoxy(2,3);
write('    Popis funkcnich klaves : ');
Textcolor(10);write('<K>         ');
Textcolor(14);
write('Ukonceni emulace :  ');end
else begin gotoxy(3,3);
write('    Navrat bez ulozeni :  ');
Textcolor(10);write(' Esc');
Textcolor(14);write(sp:9,
'Navrat s ulozenim  : ');end;
Textcolor(10);write('Ctrl-A ');
end;

procedure Init(u:byte);
label 2;
var ch:byte;f:text;
procedure Default;
begin Textbackground(0);Textcolor(14);
window(50,4,77,18);clrscr;
Textbackground(green);write(sp:28);
write(' N A S T A V E N E  HODNOTY ');
write;write(sp:28);write;
writeln;writeln;Textbackground(0);
write('  Seriovy port     :  ');
Textcolor(10);write(p[comnr]);
Textcolor(14);writeln;
write('  Rychlost prenosu :  ');
Textcolor(10);write(bd:4);Textcolor(14);
writeln;write('  Parita           : ') ;
Textcolor(10);write(x[par]);
Textcolor(14);writeln;
write('  Pocet dat. bitu  :  ');
Textcolor(10);write(db:4);Textcolor(14);
writeln;write('  Pocet stop bitu  :  ');
Textcolor(10);write(sb:4);Textcolor(14);
writeln;write('  Prvni volba      :  ');
Textcolor(10);write((yk):4);
Textcolor(14);
end;


procedure Save;
begin assign(f,'EMU89.CFG');rewrite(f);
  writeln(f,comnr);writeln(f,bd);
  writeln(f,par);writeln(f,db);
  writeln(f,sb);writeln(f,yk);
  writeln(f,drv);close(f);
end;

procedure MainWindow(m:strarr;n,l:byte);
var i,li:byte;
begin li:=23;
  if l=0 then begin li:=3;
Textbackground(0);window(4,4,40,18);
clrscr;Textbackground(green);
write(sp:38);write('    Z M E N A  ',
'P A R A M E T R U     ');write(sp:36);
end;window(4,9,40,(9+n));
Textbackground(green);gotoxy(li,1);
write(m[1]);gotoxy(li,wherey+1);
Textbackground(0);
for i:=2 to n+1 do begin write(m[i]);
gotoxy(li,wherey+1);end;gotoxy(li,1);
end;

procedure Pdspf
(m:strarr;ds,o:integer;var res:word);
begin MainWindow(m,o,1);
   repeat repeat until keypressed;
c:=readkey;if c=#0 then Arrows(m,ds,1,o)
              else begin case ord(c) of
  27:exit;13:begin res:=wherey;exit;end;
              else write(#7);end;end;
   until false;
end;


procedure Speed;
begin MainWindow(e,7,1);
   repeat repeat until keypressed;
c:=readkey;if c=#0 then Arrows(e,5,1,7)
           else begin case ord(c) of
   27:exit;13:begin case wherey of
    1:bd:=9600;2:bd:=4800;3:bd:=2400;
    4:bd:=1200;5:bd:= 600;6:bd:= 300;
    7:bd:= 150;8:bd:= 110;else bd:=9600;
    end;exit;end;else write(#7);end;end;
   until false;
end;

begin ch:=1;if u=0 then begin
 {$I-}assign(f,'EMU89.CFG');
      reset(f);close(f);{$I+}
      if IOResult=0 then begin
 assign(f,'EMU89.CFG');reset(f);
 readln(f,comnr);readln(f,bd);
 readln(f,par);readln(f,db);
 readln(f,sb);readln(f,yk);
 readln(f,drv);close(f);ch:=0;end;end;
      if ch=1 then begin
      if u=0 then begin comnr:=1;
bd:=9600;par:=3;db:=8;sb:=1;yk:=1;
drv:='C';end;Textbackground(white);
 window(1,1,80,25);clrscr;Textcolor(14);
 gotoxy(25,2); write('Emulator ',
 'terminalu  E M U 8 9');Bottom(0);
2:Default;MainWindow(m,5,0);
  repeat repeat until keypressed;
c:=readkey;if c=#0 then Arrows(m,18,1,5)
          else begin case ord(c) of
    27:begin if u=0 then Save;exit;end;
    1 :begin Save;exit;end;
    13:begin case wherey of
 1:begin Pdspf(p,4,1,comnr);goto 2;end;
 2:begin Speed;goto 2;end;
 3:begin Pdspf(x,5,2,par);goto 2;end;
 4:begin Pdspf(d,6,1,db);
         db:=db+6;goto 2;end;
 5:begin Pdspf(b,10,1,sb);goto 2;end;
 6:begin Pdspf(t,13,2,yk);goto 2;end;
 else begin write(#7);goto 2;end;end;
 end;    else write(#7);end;end;
  until false;end;
end;

begin
r:=q+q;term:=ter1;z:=u+u;v:=h+h;zac:=0;
s[1]:='1. Emulace terminalu CM7202';
s[2]:='2. Emulace terminalu CM7209';
s[3]:='3. Zmena  parametru prenosu';
m[1]:=' Seriovy  port    ';
m[2]:=' Rychlost prenosu ';
m[3]:=' Parita           ';
m[4]:=' Pocet dat. bitu  ';
m[5]:=' Pocet stop bitu  ';
m[6]:=' Prvni volba      ';
p[1]:='COM1';p[2]:='COM2';e[1]:=' 9600';
e[2]:=' 4800';e[3]:=' 2400';
e[4]:=' 1200';e[5]:='  600';
e[6]:='  300';e[7]:='  150';
e[8]:='  110';x[1]:='licha';
x[2]:=' suda';x[3]:='zadna';
d[1]:='7 bitu';d[2]:='8 bitu';
b[1]:='1  stopbit';b[2]:='2 stopbity';
t[1]:='      CM 7202';
t[2]:='      CM 7209';
t[3]:=' Inicializace';
GetIntVec(ComInt,altcom);
```

```
exitsave:=exitproc;exitproc:=@myexit;
CheckBreak:=false;Origmode:=Lastmode;
Textmode(CO80);5:Textbackground(white);
window(1,1,80,25);clrscr;
Textbackground(blue);window(4,2,77,16);
clrscr;Textbackground(green);
Textcolor(14);gotoxy(10,3);
write(r,v,v,v,v,v,v,v,v,v,v,v,v,v,v,v,
v,v,v,v,v,v,v,v,v,v,v,v);gotoxy(10,4);
write(r,sp:15,r,v,v,v,h,q,v,v,v,r,sp:3,
r,sp:14,r);gotoxy(10,5);write(r,sp:15,
r,sp:7,q,sp:6,r,sp:3,r,sp:14,r);
gotoxy(10,6);write(r,sp:15,r,sp:7,q,
sp:6,r,sp:3,r,sp:14,r);gotoxy(10,7);
write(r,sp:15,r,sp:7,q,sp:6,r,sp:3,r,
sp:14,r);gotoxy(10,8);write(r,z,z,z,z,
z,sp:5,r,sp:14,r,sp:3,r,sp:14,r);
gotoxy(10,9);write(r,sp:15,r,sp:14,r,
sp:3,r,sp:14,r);gotoxy(10,10);write(r,
sp:15,r,sp:14,r,sp:3,r,sp:14,r);
gotoxy(10,11);write(r,sp:15,r,sp:14,r,
sp:3,r,z,z,z,z,z,z,z,z,r);gotoxy(10,12);
write(r,z,z,z,z,z,z,z,z,z,z,z,z,z,z,z,z,
z,z,sp,'(c) 1989 - PKSoft');Bottom(1);
window(4,18,77,21);Textcolor(143);
gotoxy(34,2);write('Okamzik');
if zac=0 then begin Init(0);
        Inc(zac);yl:=yk;goto 5;end;
Textcolor(12);Textbackground(0);
gotoxy(26,yl);write(s[wherey]);
Textbackground(white);case yl of
1:begin gotoxy(26,2);write(s[2]);
gotoxy(26,3);write(s[3]);gotoxy(26,1);
end;2:begin gotoxy(26,3);write(s[3]);
gotoxy(26,1);write(s[1]);gotoxy(26,2);
end;3:begin gotoxy(26,1);write(s[1]);
gotoxy(26,2);write(s[2]);gotoxy(26,3);
end;end;repeat i:=0;if zac=1 then begin
  repeat Inc(i);until keypressed or
  (i=15000);if i=15000 then c:=#13 else
  c:=readkey;Inc(zac);end
  else begin repeat until keypressed;
  c:=readkey;end;
      if c=#0 then Arrows(s,27,0,2)
              else begin case ord(c) of
1: exit;13: begin case wherey of
2: begin yl:=2;term:=ter2;Emulace;end;
3: begin yl:=3;Init(1);end;
else begin yl:=1;term:=ter1;Emulace;end;
     end;goto 5;end;
48: begin yl:=1;term:=ter1;Emulace;
     goto 5;end;
50: begin yl:=2;term:=ter2;Emulace;
     goto 5;end;
51: begin yl:=3;Init(1);goto 5;end;
107,75:begin yl:=wherey;Keys;goto 5;end;
else write(#7);end;end;until false;
end.
```

PRIPOJENIE TLAČIARNE

K6314 K PMD 85–2 A SHARP MZ800

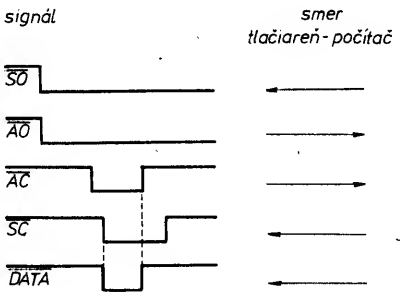
Ing. Peter Synek, *Jána Smreka 12, 841 07 Bratislava*

Dostal som sa pred úlohu pripojiť osobné mikropočítače typu PMD 85–2 a Sharp MZ800 k tlačiarni Robotron K6314. Každé z týchto zariadení je vybavené iným typom interfejsu. Tlačiareň má interfejs typu IFSP (u nás sa označuje IRPR), mikropočítač Sharp MZ800 má interfejs typu Centronics a PMD 85–2 využíva mód 1 integrovaného obvodu MHB 8255A s podporou handshake.

Popis interfejsu IRPR (obr. 1)

Výmena informácií (D0 ÷ D7)n sa vykonáva za pomoci signálov ACn a SCn v režime žiadosť – odpoveď. Signálom ACn žiada tlačiareň o vyslanie dát. Počítač platnosť dat potvrdzuje signálom SCn.

Počítač môže vyslať data a signál SCn iba za podmienky, keď tlačiareň vysiela signál ACn. Tlačiareň môže ukončiť ACn za podmienky, keď počítač vysiela signál SCn.



Obr. 1. Signály interfejsu IRPR (903–1)

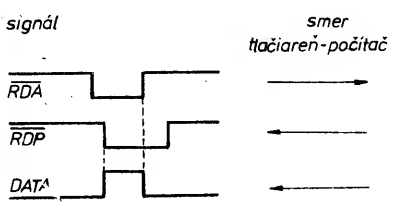
Popis intefejsu Centronics – Sharp MZ 800 (obr. 2)

Bližší popis nájdu záujemci v [1].

Porovnaním s intefejsom IRPR možno napísať:

RDAn = ACn
RDPn = SCn

Data sú vzájomne invertované.



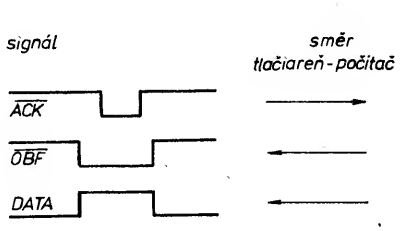
Obr. 2. Signály interfejsu Centronics Sharp (903–2)

Popis interfejsu PMD85–2 (obr. 3)

Využíva sa tu mód č. 1 integrovaného obvodu MHB8255A a bližší popis možno nájsť v príslušnej dokumentácii k tomuto obvodu (napríklad v [6]). V krátkosti možno zhrnúť: Podmienka vyslania informácie je, že signál ACKn musí byť v stave log.1. Ak tento signál invertujeme, môžeme povedať, že invertovaný signál ACKn v log.0 znamená žiadosť tlačiarne o vyslanie informácie, t. j. signál ACn. Porovnaním s IRPR teda možno konštatovať:

OBFn = SCn
ACK = ACn

Data sú vzájomne invertované.



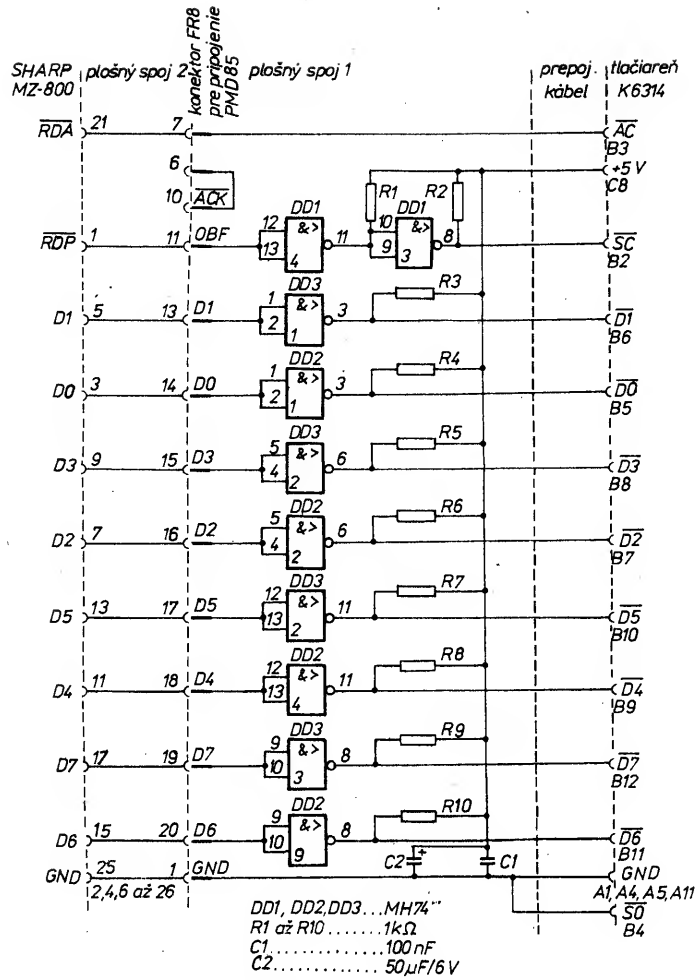
Obr. 3. Signály interfejsu PMD-85 (903–3)

Koncepcia a riešenie prepojenia

Na základe predošlých konštatovaní bola zvolená nasledovná koncepcia riešenia. Bola vyhotovená jedna doska plošného spoja zakončená konektorom FRB, ktorým sa pripája priamo do PMD 85–2. Z druhej strany je vedený kábel zakončený konektorom pre pripojenie tlačiarne (konektor sa dodáva s tlačiarňou). Pre pripojenie osobného mikropočítača Sharp MZ 800 bola zrealizovaná „redukcia", prevádzajúca signály z priameho konektora počítača Sharp na konektor FRB tak, aby sa tlačiareň pripájala cez tento konektor FRB rovnako ako ku počítaču PMD 85–2.

Realizácia celého pripojenia je zrejmá zo schémy na **obr. 4.** Obrazec plošných spojov je na **obr. 5.**

Hradlá integrovaných obvodov DD2 a DD3 slúžia na invertovanie výstupných datových signálov z mikropočítačov. Hradlá



Obr. 4. Schéma pripojenia (903–4)

DD1.3 a DD1.4 slúžia na výkonové prispôsobenie signálu OBFn z počítača PMD 85–2, ktorý je vedený priamo z MHB8255A bez výkonového zosilnenia. Vstupné obvody IRPR tlačiarne sú riešené odporovým prispôsobením s rezistormi malých odporov a priame pripojenie by mohlo viesť k poškodeniu MHB8255A. V prípade počítača Sharp MZ800 vlastne tieto hradlá nemajú význam, pretože všetky jeho výstupy sú výkonovo prispôsobené. Signál ACn sa z tlačiarne privádza priamo na vývod č. 7 konektoru FRB, odkiaľ je v prípade počítača Sharp MZ800 vedený ako signál RDAn priamo do počítača. V prípade PMD 85–2 treba tento signál invertovať. Na to sa využíva invertor, ktorý je na konektore FRB v PMD85–2 vyvedený. Vývod č. 7 je vstupom tohoto invertora, jeho výstup je na vývode č. 6, odkiaľ je signál vedený na vývod číslo 10 ako signál ACKn.

Signál SOn je natrvalo pripojený na log.0.

Napájacie napätie +5 V pre integrované obvody je použité z tlačiarne cez nevyužitý vývod č. C8.

Realizácia „redukcie" pre pripojenie Sharp MZ800 spočíva iba v pripevnení vhodných konektorov na spoločnú doštičku a ich vzájomnom prepojení drátovými spojmi. Prepoje sú jasné zo schémy **(obr. 4).**

Oživenie a uvedenie do chodu by nemalo robiť problémy vzhľadom na jednoduchosť zapojenia. Nesmie sa však zabudnúť v interfejsu IRPR tlačiarne vyviesť na príslušný nevyužitý vývod (u nás C8) napájacie napätie +5 V).

Zoznam súčiastok

DD1, DD2, DD3	MH7438
R1 ÷ R10	1kΩ, TR 211
C1	100nF, TK 783
C2	50 μF/6V, TE 981
konektor FRB	TY5122011/20
konektor FRB pre Sharp MZ800	TX5112013/20
konektor	WK46580 (skrátený na 2×13 vývodov) pre Sharp MZ800

Literatúra

1 Centronics. Amatérské rádio A 10/1984, str. 382.

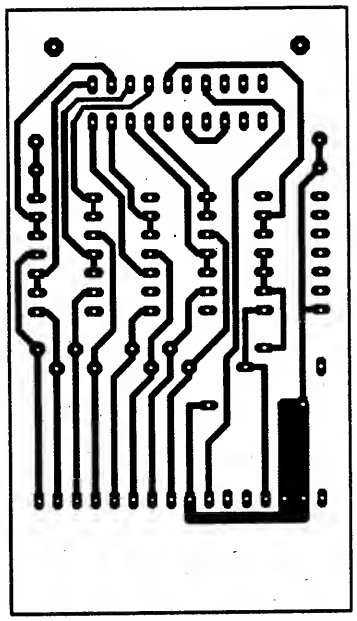
2 Systém malých elektronických počítačov SMEP, Abecednočíslicový displej SM 72–00. Technická dokumentácia, Vydavatelství Novinár Praha pre Teslu Orava.

3 Technický popis modulu UKPR–8. Technická dokumentácia TESLA.

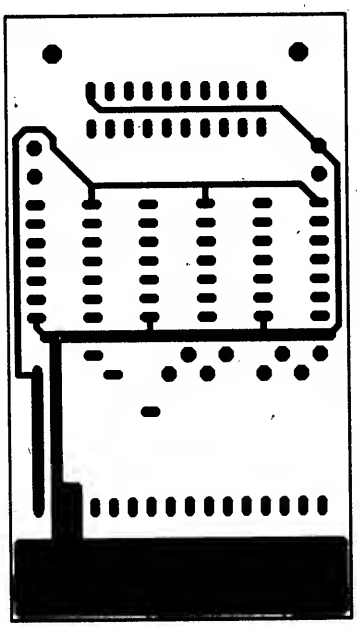
4 MZ 800. Owner's manual, Sharp.

5 PMD 85. Učebný materiál kurzu „Basic G", Datasystém.

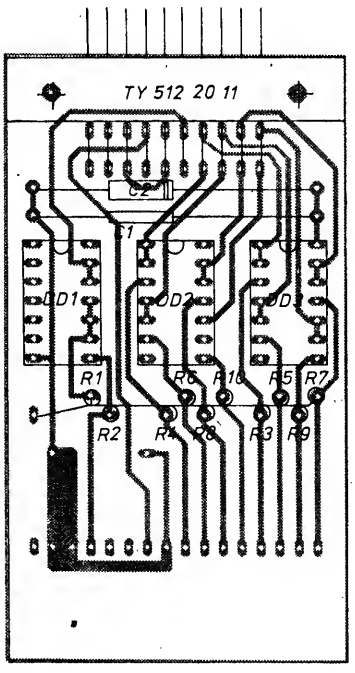
6 *Kočiš, I., Šulko, I.:* Mikroprocesory a mikropočítače. ALFA, Bratislava – 1986.



Obr. 5. Obrazec plošných spojov zo strany spojov na doske Y504 (903–5)



Obr. 6. Obrazec plošných spojov zo strany súčiastok na doske Y504 (903–6)



Obr. 7. Rozmiestnenie súčiastok na doske Y504 (903–7)

GENERÁTOR TISKOVÝCH SESTAV GTS

Ing. Ivo Křepinský, *Karlovarská 5, 301 12 Plzeň*

Generátor tiskových sestav slouží k ulehčení práce při vytváření programů pro tisk. Tvoří jej soubor procedur v Turbo Pascalu, které je nutno připojit k programu.

Tisková sestava se tiskne podle předlohy vytvořené předem textovým editorem. V předloze jsou vyznačena pole pro tisk hodnot. Do těchto polí se při běhu programu doplňují konkrétní údaje a výsledná sestava se tiskne.

Tiskovou sestavu je možno nejen vytisknout na tiskárně, ale i na jiném zařízení (terminál), případně ji uložit do souboru.

Výhody GTS:
- jednoduchý a rychlý návrh tiskové sestavy,
- zkrácení programu pro tisk,
- snadnější ladění bez složitého vypočítávání polohy na papíru,
- oddělení grafické a obsahové složky tisku,
- možnost úprav tiskové sestavy bez zásahu do programu,
- navržená tisková sestava slouží zároveň jako část dokumentace,
- výstup je možno ve stejném tvaru posílat na libovolné výstupní zařízení nebo do souboru.

Generátor tiskových sestav je vytvořen v Turbo Pascalu v.3 a je použitelný v programech napsaných v tomto jazyku na všech počítačích, kde pracuje překladač tohoto jazyka.

Vytvoření předlohy

Předloha pro tiskovou sestavu se vytváří libovolným textovým editorem (např. WordStar). Obsahuje pevné texty sestavy v tom tvaru, jak mají vypadat na výstupu. Dále obsahuje pole pro výstup hodnot. Tato pole jsou označena znakem "_" (podtržítko). Délka pole je určena počtem podtržítek.

Pole musí být na jedné řádce, nesmí být rozděleno na více řádek. Není možno umístit dvě pole těsně za sebou bez oddělení jiným znakem.

Soubor obsahující předlohu tiskové sestavy musí mít standardní příponu ".TS".

Použití GTS

Vyplňování hodnot

Program nejprve musí přečíst ze souboru předlohu tiskové sestavy (procedura TSREAD). Předloha setrvává v paměti a je možno ji používat, dokud není přečtena jiná.

Po přečtení předlohy je možno postupně vyplňovat pole pro hodnoty. Aktivní pole je určeno polohou ukazovátka. Po přečtení předlohy, po tisku sestavy a po užití procedury TSFIRST je ukazovátko nastaveno na první pole tiskové sestavy. Každý zápis hodnoty (procedura TSPRINT) posunuje ukazovátko na další pole. Procedura TSSKIP umožňuje přeskočit pole bez vyplnění.

Ovládání výstupního souboru

Výstup je možno posílat do libovolného souboru, který je otevřený v Turbo Pascalu. Při použití standardních identifikátorů (CON:, LST:) je možno výstup posílat i na výstupní zařízení.

Před všemi výstupy je nutno soubor otevřít (procedura TSOPEN), po ukončení uzavřít (procedura TSCLOSE).

Všechny výstupy (procedury TSOUTPUT a TSWRITE) se posílají do otevřeného souboru.

Použití generátoru v programu

Na počátku programu je nutno nadefinovat konstantu TSLEN, která udává maximální počet znaků největší předlohy, která bude v programu použita.

Generátor se k programu připojí direktivou {$I GTS}.

Procedury pro práci s generátorem

TSOPEN(Soubor:string[14])
– otevírá výstupní soubor,
– tato procedura musí být použita před všemi výstupy (TSOUTPUT a TSWRITE),
– může být uvedeno libovolné jméno souboru (tiskárna je LST:, terminál CON:),
– všechny výstupy prováděné procedurami TSOUTPUT a TSWRITE se provádějí do otevřeného souboru.

TSCLOSE
– zavírá výstupní soubor,
– je nutno použít, pokud je výstupním souborem diskový soubor, jinak se data nezapíší.

TSREAD(Soubor:string[14])
– čte předlohu tiskové sestavy ze zadaného souboru,
– jméno souboru se zadává bez přípony – ta je vždy .TS,
– ukazovátko se nastaví na první pole sestavy,
– předloha sestavy zůstává v paměti až do přečtení jiné předlohy.

TSPRINT(Hodnota:string[255])
– zapisuje hodnotu do tiskové sestavy,
– zápis se provádí vždy do současného pole,
– po zápisu se ukazovátko nastaví na následující pole,
– při zápisu se začíná od prvního znaku výstupního pole,
– znaky hodnoty přesahující délku pole se odříznou,
– zbývající znaky do délky pole se doplní mezerami,
– zápis se provádí pouze do paměti, výstup se provede až procedurou TSOUTPUT.

TSSKIP
– přeskakuje současné pole,
– ukazovátko se nastaví na následující pole.

TSFIRST
– nastavuje ukazovátko na první pole sestavy.

TSOUTPUT
– zapíše kompletní sestavu do výstupního souboru,
– místo nevyplněných polí se tisknou mezery,
– po skončení zápisu se nastaví ukazovátko na první pole sestavy a je možno znovu vyplňovat pole.

TSWIRE(Retezec:string[255])
– zapíše řetězec do výstupního souboru,
– slouží pro nestandardní výstupy a pro ovládání tiskárny (např. odstránkování),
– řetězec se do výstupního souboru zapíše přesně v zadaném tvaru,
– zápis do výstupního souboru se provádí ihned.

Poznámka: Pokud mají procedury parametry, jsou všechny typu řetězec. Procedury nevracejí žádné hodnoty.

Popis realizace GTS

Generátor tiskových sestav je navržen jako soustava procedur. Pro přehlednost a názornost jsou procedury dále členěny na menší podprogramy.

Názvy všech objektů (typů, proměnných a podprogramů), které jsou deklarovány jako globální, ale jsou určeny pouze pro GTS, začínají znaky "TS_", aby byla vyloučena možnost kolize. Názvy všech objektů, které je možno v programu využívat, začínají znaky "TS".

Pomocné proměnné

TS_M:array[0 .. TSLEN]of char
pole pro předlohu tiskové sestavy, naplňuje se při TSREAD; výstupní pole jsou označena nastavením nejvyššího bitu na 1.

TS_Ptr:Integer
ukazatel do pole TS_M na akutální znak; obsahuje polohu, kde se bude číst nebo zapisovat.

TS_Soubor:text
výstupní soubor.

Pomocné procedury a funkce

TS_Init
inicializace ukazatele TS_Ptr.

TS_Put (Znak:char)
zápis znaku do pole TS_M a zvýšení ukazatele TS_Ptr.

TS_Get:char
čtení znaku z pole TS_M a zvýšení ukazatele TS_Ptr.

TS_Dirget:char
čtení znaku z pole TS_M bez zvýšení ukazatele TS_Ptr.

TS_IsF(Znak:char)boolean
určuje, zda je daný znak součástí pole pro výstup hodnoty.

TS_SetF(Znak:char):char
vrací zadaný znak jako součást pole pro výstup hodnoty.

TS_GetF(Znak:char):char
vrací zadaný znak bez označení jako součást pole pro výstup hodnoty.

Poznámka:
Realizace je navržena tak, aby bylo možno provést některé změny. Pomocí procedur a funkcí je oddělen problém uschování předlohy tiskové sestavy a problém označení polí pro výstup hodnoty.

Popis algoritmu hlavních procedur

TSOPEN
– přiřazení názvu souboru k TS_Soubor,
– otevření pro zápis.

TSCLOSE
– zavření souboru.

TSREAD – lokální proměnné:
Soubor:file – soubor předlohy.
Sektor:array | 0 . . 127 | of char – oblast pro čtení sektoru.
– přiřazení názvu předlohy k Soubor a otevření pro čtení,
– inicializace TS_Ptr,
– čtení sektoru ze Soubor až do konce souboru:
  – pro každý sektor,
    – Převod znaku '_' na vnitřní označení výstupního pole,
    – uložení znaku do paměti,
– zavření souboru předlohy,
– přidání ^Z na konec,
– nastavení prvního pole (TSFIRST).

TSFIRST
– inicializace TS_Ptr,
– vyhledání prvního znaku s označením výstupního pole.

TSSKIP
– přeskočení znaků s označením výstupního pole
– vyhledání následujícího znaku s označením výstupního pole.

TSPRINT
– od současné polohy se hodnota zapisuje až do délky pole,
– zbývající znaky hodnoty se ignorují,
– pokud není vyčerpána délka pole, doplní se mezerami,
– skok na další pole (TSSKIP).

TSOUTPUT
– inicializace TS_Ptr,
– celá sestava, znak po znaku až do ^Z, se zapíše do souboru,
– nastaví se první pole (TSFIRST).

Návrh úprav programu

Zjednodušení programu:
– většinu pomocných podprogramů je možno zrušit a jejich činnost přenést na místo jejich volání – program se tím zkrátí,
– výstup je možné omezit pouze na tiskárnu; pak je možno zrušit procedury TSOPEN a TSCLOSE a zjednodušit výstupy.

Vylepšení programu:
– přidělovat paměť pro předlohu tiskové sestavy dynamicky,
– vytvořit zásobník předloh v paměti a tím umožnit opakované použití předlohy bez jejího nového čtení,
– umožnit v proceduře TSSKIP přeskakování polí nazpátek a přeskakování o zadaný počet polí,
– zavést různé formátování výstupu hodnot do polí sestavy (např. zarovnání vpravo, na daný počet desetinných míst) (většina těchto úprav však s sebou přináší značné zvětšení programu bez patrných výhod pro programátora).

Realizace *v jiných jazycích:*

Navržený algoritmus GTS je možno realizovat i v jiných programovacích jazycích téměř shodně.
Pro počítače, které nemají diskovou paměť, by bylo možné algoritmus upravit tak, aby pracoval s předlohou tiskové sestavy zadanou jako data v programu. Tím se však odstraňuje jedna z hlavních výhod GTS – jednoduché a názorné vytváření předlohy.

Příklad použití

Použití GTS je předvedeno na jednoduchém příkladu. Program tiskne seznam pracovníků z diskového souboru.
Každý list sestavy se skládá z hlavičky a výpisu údajů o několika zaměstnancích. Program využívá dvě předlohy pro tyto dvě části tiskové sestavy – jednu pro hlavičku a jednu pro údaje o jednom zaměstnanci. Na každou stranu se pak tiskne nejprve hlavička s doplněným datem a číslem strany a pak daný počet zaměstnanců (pro každého se doplňuje jeho pořadové číslo a další údaje ze souboru).
V příloze je uveden výpis programu, předloha obou částí sestavy (obr. 1) a příklad sestavy (obr. 2).

```
Strana: ____                                        Datum: ________
                          SEZNAM PRACOVNIKU
================================================================================

Cislo: _____   Prijmeni:  ____________________
               Jmeno:     ____________________
               Datum narozeni: ________   Misto: ________________
--------------------------------------------------------------------------------
```

Obr. 1. Předloha obou částí sestavy (930–1)

```
Strana: 1                                           Datum: 01-08-89
                          SEZNAM PRACOVNIKU
================================================================================

Cislo:     1   Prijmeni:        Novak
               Jmeno:           Josef
               Datum narozeni:  01-01-40      Misto:  Plzen
--------------------------------------------------------------------------------

Cislo:     2   Prijmeni:        Soukup
               Jmeno:           Jiri
               Datum narozeni:  10-02-56      Misto:  Praha
--------------------------------------------------------------------------------
```

Obr. 2. Příklad sestavy (930–2)

Výpis 1. *Generátor tiskových sestav* (930–V1)

```
{*****************************
 *                           *
 *          GTS              *
 * generator tiskovych sestav *
 *                           *
 *     Ing. Ivo Krepinsky    *
 *       (c) 1988,1989       *
 *                           *
 *****************************

 Popis:
     Soubor obsahuje procedury
 pro tisk sestav.

 Pouziti:
     Soubor se k programu pri-
 dava direktivou $I TS. Na za-
 catku programu musi byt prove-
 deno nastaveni konstanty TSLEN
 - maximalni delka tiskove se-
 stavy v bytech }
```

```
type
 TS_Str14=string[14];
 TS_Str255=string[255];

var
 TS_M:array [0..TSLEN] of char;
 TS_Ptr:Integer;
 TS_Soubor:text;

procedure TS_Init;
 begin
  TS_Ptr:=0;
 end;

procedure TS_Put(Znak:char);
 begin
  TS_M[TS_Ptr]:=Znak;
  if TS_Ptr<TSLEN then
   TS_Ptr:=TS_Ptr+1;
 end;

function TS_Get:char;
 begin
  TS_Get:=TS_M[TS_Ptr];
  if TS_Ptr<TSLEN then
   TS_Ptr:=TS_Ptr+1;
 end;

function TS_DirGet:char;
 begin
  TS_DirGet:=TS_M[TS_Ptr];
 end;

function TS_IsF(Znak:char):
                        boolean;
 begin
  TS_IsF:=Znak)#127;
 end;

function TS_SetF(Znak:char):
                          char;
 begin
  TS_SetF:=Chr(Ord(Znak)or $80);
 end;
{vypis seznamu pracovniku}
{ukazkovy program pouziti GTS}

const
 TSLEN=262;

{$I GTS}

type
 Str8=string[8];

procedure Hlavicka
    (Strana:integer;Datum:Str8);
 var
  Cislo:string[4];
 begin
  if Strana)1 then TSWRITE(^L);
  TSREAD('TEST1');
  Str(Strana,Cislo);
  TSPRINT(Cislo);
  TSPRINT(Datum);
  TSOUTPUT;
  TSREAD('TEST2');
 end;


var
 Datum:Str8;
 Soubor:string[14];
 Data:text;
 Prijmeni:string[20];
 Jmeno:string[15];
 Nar:string[8];
 Misto:string[15];
 Strana,Cislo:integer;
 RetCis:string[5];

begin
 Write('Datum: ');
 Readln(Datum);
 Write('Vystupni soubor: ');
 Readln(Soubor);
 TSOPEN(Soubor);
 Assign(Data,'TEST.DAT');
 Reset(Data);
 Strana:=1;
 Cislo:=0;
 while not EOF(Data) do
  begin
   if (Cislo mod 4)=0 then
    Hlavicka(Strana,Datum);
   Readln(Data,Prijmeni,Jmeno,
                        Nar,Misto);
   Cislo:=Cislo+1;
   Str(Cislo:5,RetCis);
   TSPRINT(RetCis);
   TSPRINT(Prijmeni);
   TSPRINT(Jmeno);
   TSPRINT(Nar);
   TSPRINT(Misto);
   TSOUTPUT;
  end;
 Close(Data);
 TSWRITE(^L);
 TSCLOSE;
end.
  while TS_DirGet()^Z do
   Write(TS_Soubor,
                TS_GetF(TS_Get));
  TSFIRST;
 end;

procedure TSWRITE(Value:
                        TS_Str255);
 begin
  Write(TS_Soubor,Value);
 end;

procedure TSOPEN(Spec:TS_Str14);
 begin
  Assign(TS_Soubor,Spec);
  Rewrite(TS_Soubor);
 end;

procedure TSCLOSE;
 begin
  Close(TS_Soubor);
 end;
```

Výpis 2. Ukázkový program použití GTS (930–V2)

```
function TS_GetF(Znak:char):
                              char;
 begin
  TS_GetF:=Chr(Ord(Znak) and
                              $7F);
 end;

procedure TSFIRST;
 begin
  TS_Init;
  while not TS_IsF(TS_DirGet) do
   if TS_Get=^Z then Exit;
 end;

procedure TSSKIP;
 var
  Znak:char;
 begin
  while TS_IsF(TS_DirGet) do
   Znak:=TS_Get;
  while not TS_IsF(TS_DirGet) do
   if TS_Get=^Z then Exit;
 end;

procedure TSREAD(Spec:TS_Str14);
 var
  Soubor:file;
  Sektor:array [0..127] of char;
  I:integer;
 begin
  Assign(Soubor,Spec+'.TS');
  Reset(Soubor);
  TS_Init;
  while not EOF(Soubor) do
   begin
    BlockRead(Soubor,Sektor,1);
    for I:=0 to 127 do
     begin
      if Sektor[I]='_' then
       Sektor[I]:=TS_SetF(' ');
      TS_Put(Sektor[I]);
     end;
   end;
  Close(Soubor);
  if Sektor[I]<>^Z then
   TS_Put(^Z);
  TSFIRST;
 end;

procedure TSPRINT(Value:
                          TS_Str255);
 var
  I:integer;
 begin
  for I:=1 to Length(Value) do
   if TS_IsF(TS_DirGet) then
    TS_Put(TS_SetF(Value[I]));
  while TS_IsF(TS_DirGet) do
   TS_Put(TS_SetF(' '));
  TSSKIP;
 end;

procedure TSOUTPUT;
 begin
  TS_Init;
```

HARDWAROVÁ NÁSOBIČKA MH102

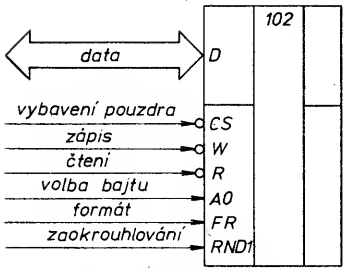
Ing. Petr Kandera, *Krasnoarmejců 2, 704 00 Ostrava 3*

Článek se zabývá použitím hardwarové násobičky MH102 v osmibitových mikropočítačích. Při praktických aplikacích mikropočítačů např. v měřicí technice se dostává do popředí problém aritmetického zpracování naměřených dat. Čistě programové řešení výpočtu je obvykle časově náročné, a proto se k jeho urychlení používají aritmetické koprocesory (např. I 8231, I 8232 apod.), které se však obtížně shánějí a jsou drahé. Přitom v řadě aplikací vystačíme kromě součtu a rozdílu pouze s násobením, které ovlivňuje celkovou dobu výpočtu rozhodující měrou.

Urychlit výpočet násobení lze použitím hardwarové násobičky s integrovaným obvodem MH102 k.p. Tesla Rožnov. Tento programovatelný periferní obvod, vhodný pro použití v systémech s mikroprocesorem typu 8080, je vyrobený technologií I^2L a představuje úplnou násobičku 8 × 8 bitů s možností zaokrouhlení a práce se znaménkem v pevné řádové čárce.

Principiální zapojení násobičky je na **obr. 1.** Obvod MH102 obsahuje 6 řídicích vstupů: CS, W, RD, A0, FR a RND1, a 8 vstupů/výstupů datové sběrnice. Vstup CS je určen k vybavení obvodu a vstupy W a RD řízení zápisu resp. čtení dat. Logická úroveň na vstupu A0 určuje při zápisu prvního a druhého činitele a při čtení vyšší nebo nižší bajt výsledku, vstup FR řídí formát násobení a RND1 zaokrouhlování podle funkční tabulky **Tab. 1.**

Při praktické realizaci je nutné připojit obvod na datovou sběrnici mikropočítače přes



Obr. 1. Principiální zapojení násobičky (913–1)

FR	A0	RND1	Funkce
X	H	X	Vyšší bajt
X	L	X	Nižší bajt
H	X	X	Formát sedmibitový se znaménkem
L	X	X	Formát osmibitový
H	H	H	Zaokrouhlení D_{01} = D'9 + přenos (D'7 + 1)
L	H	H	Zaokrouhlení D_{01} = D'9 + přenos (D'8 + 1)
X	X	L	Bez zaokrouhlení

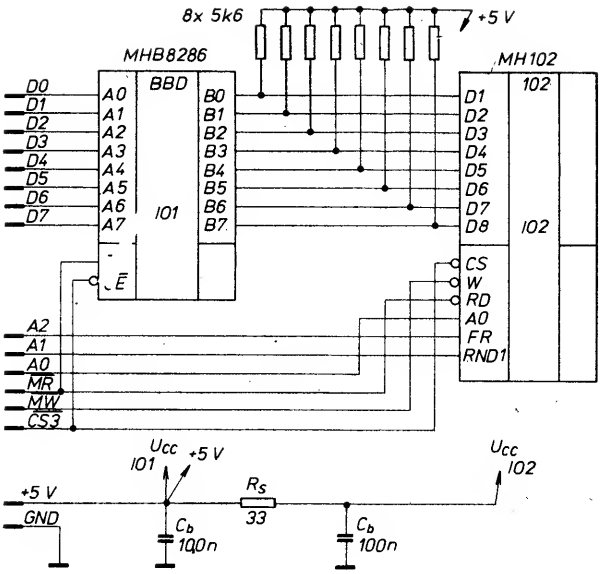
***Tab. 1.** Funkční tabulka (913–T1)*

stykový obvod, např. 8255, 8286 nebo 3216. Řídicí signály lze odvodit z adresové a řídicí sběrnice, např. pro zápis a čtení signály IOWn a IORn, případně MWn a MRn mikroprocesoru 8080 a pro řízení násobení signály adresové sběrnice. Signál CS pro vybavení obvodu odvodíme adresovým dekodérem (3205). Na **obr. 2** je uvedeno konkrétní zapojení úplné násobičky pro základní systém mikropočítače s mikroprocesorem 8080, představovaný mikropočítačem PMI-80. Toto zapojení je realizováno s ohledem na maximální jednoduchost a minimální nároky na programovou podporu. Pro připojení na datovou sběrnici je využit oboustranný budič sběrnice 8286, řízený signály MRn a CS3n. Signál MRn řídí směr přepisu dat a signál CS3n výběr pouzdra (i u MH102). Tento signál je získán z adresového dekodéru 3205 v PMI-80 a je původně určen k řízení stránek paměti. Násobička je tedy adresována jako sektor paměti a nikoliv jako V/V obvod, což kromě větší rychlosti přináší i výhodu většího instrukčního souboru pro práci s daty. Zápis a čtení dat obvodu MH102 řídí signály MWn a MRn a tři nejnižší bity adresované sběrnice specifikují násobení. A0 nastavuje nižší a vyšší bajt, A1 určuje zaokrouhlování a A2 formát násobení. Protože obvod MH102 je vyroben technologií I^2L, jsou pro přizpůsobení datových signálů určeny rezistory R (5,6kΩ). Napájení obvodu je proudové z 5V přes sériový rezistor R_s (33Ω), napájení je u každého obvodu blokováno kondenzátory C_b (100 nF).

Programová obsluha násobičky je jednoduchá. Ze zapojení a funkční tabulky sestavíme adresy pro zápis a čtení vyššího a nižšího bajtu pro jednotlivé typy násobení. Konkrétně pro případ na **obr. 2** platí: signál CS3n vymezuje v PMI-80 adresovou oblast 0C00H až 0FFFH a volbou spodních tří bitů zajistíme požadovanou funkci. Např. pro násobení 8×8 bitů bez zaokrouhlení musí být A2 = = FR = L a A1 = RND1 = L a pro nižší bajt (první činitel) A0 = L a vyšší bajt (druhý činitel) A0 = H. Pro podpogram vystačíme pouze s instrukcemi pro zápis a čtení dat z paměti. Např. podprogram pro násobení dvou bajtů uložených v registrech H a L s výsledkem v párovém registru HL vypadá takto:

```
MUX:  SHLD 0C00H  : Uložení činitelů
      LHLD 0C00H  : Načtení výsledku
      RET
```

Obr. 2. Konkrétní zapojení úplné násobičky (913–2)



Návrh plošných spojů a podprogramy pro jednotlivé typy násobení neuvádím, protože si jistě každý uživatel přizpůsobí zapojení podle svých možností a vytvoří podprogramy podle svých požadavků. Zapojení násobičky je značně univerzální a lze jej snadno přizpůsobit pro různé typy mikropočítačů s mikroprocesory typu 8080 nebo Z80.

Literatura

[1] Polovodičové součástky 1984/85. Katalog TESLA Rožnov k.p., Rožnov pod Radhoštěm, 1983.

[2] PMI-80 – Uživatelská příručka. TESLA Piešťany k.p.

PRIPOJENIE ZAPISOVAČA

SHARP 1P16 K MIKROPOČÍTAČU SORD M5

RNDr. Peter Dubec, *Železničiarska 4, 979 01 Rím. Sobota*

Pred časom sa objevil v predaji zapisovač typu Sharp MZ-1P16, vybavený pripojením Centronics. Nakoľko počítač SORD M-5 komunikuje s tlačiarňou cez rovnaké rozhranie a originálny printer k tomuto počítaču nie je dostupný, používam uvedenú tlačiareň-zapisovač ako periférne zariadenie k tomuto počítaču.

Sharp MZ-1P16 je vlastne štvorfarebný súradnicový zapisovač riadený zabudovaným mikropočítačom, ktorý zabezpečuje činnosť v dvoch režimoch, a to v režime textovom, v ktorom komunikuje s počítačom ako tlačiareň, a v režime grafickom, v ktorom vykonáva grafické príkazy (napr. príkaz X 1,10,20 má za následok vykreslenie X-ovej osi, na ktorej bude vynesených 20 značiek vzdialených navzájom o 10 krokov). Nakoľko k zapisovaču je dodávaný podrobný návod, ktorý obsahuje popis všetkých grafických príkazov, ako aj riadiacich kódov textového režimu, nebudeme sa týmito ďalej zaoberať, ale uvedieme riešenie základných problémov, s ktorými sa stretne majiteľ počítača SORD M-5 pri pripájaní tohto zapisovača.

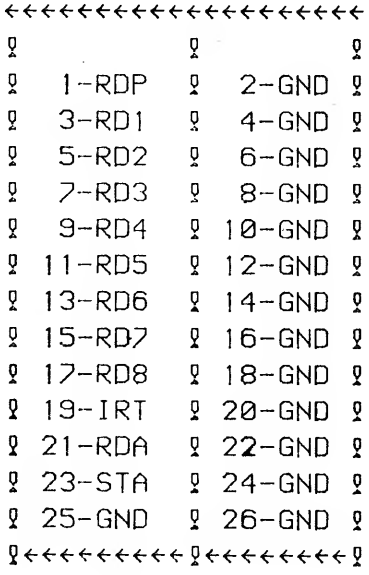
V prvom rade musíme vyriešiť napájanie. Zapisovač nemá vlastný zdroj, nakoľko výrobca predpokladal jeho pripojenie na zdroj počítača SHARP MZ-800. Užívateľ je teda nútený postaviť zdroj, ktorý dodá pri 5 V asi 2 A (kľudový odber zapisovača je len 0,5 A, ale pri zdvíhaní a sklápaní pierka dosahuje spomínané 2 A).

Po pripojení zdroja (kladný pól napájania je na vonkajšej strane konektora, kým záporný pripájame ke stredovému vodiču) a otestovaní zapisovača autotestom pristúpime k jeho pripojeniu na počítač.

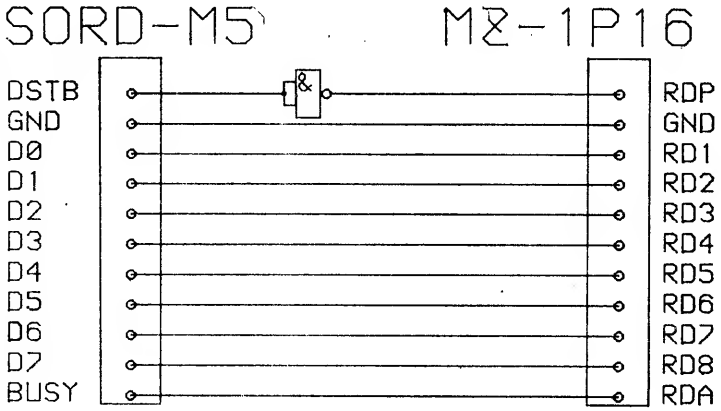
Pri pripájaní vychádzame zo zapojenia konektoru tlačiarne, ktoré je uvedené na **obr. 2.** Ďalej musíme vziať do úvahy rozdielnosti v ponímaní štandardu Centronics u oboch firiem. Táto rozdielnosť spočíva v tom, že tlačiareň považuje signál RDP (analóg STROBE) za aktívny pri vysokej logickej úrovni, kým počítač SORD M5 ho považuje za aktívny pri úrovni nízkej. Túto rozdielnosť vyriešime zapojením invertora medzi príslušné dva vývody. Výsledné zapojenie je na **obr. 1.** Ako invertor je použité jedno hradlo NAND obvodu MH7400 s prepojenými vstupmi.

Programové zabezpečenie tlače

V prípade dokonalého prepojenia konektorov (myslí sa dokonalého z hľadiska odrušenia signálu) vystačíme s rutinami obsiahnutými v monitore počítača, na ktoré sa odvoláva BASIC pri vykonávaní inštrukcií typu CLIST" PRT:" a PRINT=2 . . . , kde kanál č. 2 bol priradený tlačiarni príkazom OPEN "PRT:" AS=2 (tu sme využili príkazy jazyka BASIC-F). Týmto spôsobom možno tlačiť všetky veľké písmená, a ostatné ASCII znaky, ktorých kódy spadajú do intervalu 32 až 95. Malé písmená nezodpovedajú štandardu ASCII, pokiaľ ide o ich kódy, čo zne-



Obr. 2. Zapojenie konektoru tlačiarne (905–2)



Obr. 1. Schéma pripojenia zapisovača Sharp MZ1P15 k počítaču Sord M5 (905–1)

možňuje použitie príkazu LIST"PRT:", ktorý vysiela inštrukcie a funkcie BASIC-u reprezentované malými písmenami, ale musíme použiť inštrukciu CLIST. Kódy malých písmen sú vypísané v **tabuľke 1,** pričom za každým znakom je uvedený jeho kód v desiatkovej aj šestnástkovej sústave.

Program, ktorý priamo konvertuje text z ASCII do kódu tlačiarne SHARP, je vo **Výpise 1.** Po spustení programu príkazom RUN vyšle počítač na obrazovku hlásenie SPRÁVA a načíta text zložený maximálne z 64 znakov, ktorý vyšle na tlačiareň. Maximálnu dĺžku textu môžeme nastaviť na riadku 5 prepísaním čísla 64 na číselnú hodnotu požadovanej dĺžky textu. Na riadkoch 20 až 40 sa tvorí prevodová tabuľka kódov, pričom obsah prvej položky poľa PREV bude obsahovať kód SHARP písmena „a" a 26. položka tohto poľa kód písmena „z", pričom všetky písmená sú zoradené v abecednom poradí. Na 50. riadku sa nastavuje riadiaci bajt drajvera tlačiarne tak, aby na konci riadku vysielal len kód CR a nie CR aj LF a aby neprevádzal automatické riadkovanie. Časť programu na riadkoch 80 až 120 vykonáva konverziu reťazca načítaného na riadku 70, pričom výstup konvertovaného reťazca prebehne na riadku 130. Opätovný štart výkonnej časti programu je zabezpečený príkazom GOTO 70 na konci programu.

Štandardný drajver, obsiahnutý v monitore počítača, je možné použiť len v prípade dokonale odrušeného vedenia medzi počítačom a tlačiarňou. Ak máme problémy s prenosom (komolenie prenášaných textov a hlavne vynechávanie znakov) musíme vytvoriť program obsluhy tlačiarne v BASIC-u, ktorý zabezpečí pomalší prenos dát a tým aj vyššiu spoľahlivosť prenosu. Riešenie uvedeného problému je vo **Výpise 2.** Podprogram začínajúci návestím SOUT je náhradou drajveru Centronics, t.j. vyšle kód jedného znaku a signál STROBE privedie do aktívneho stavu, potom čaká na potvrdenie príjmu (signál READY) a nakoniec privedie signál STROBE do stavu pasívneho.

Pre úplnosť treba dodať, že datové bity rozhrania Centronics sú adresované hodnotou 64. Signál STROBE prislúcha prvému a signál READY druhému bitu vstupnovýstupného kanálu adresovaného číslom 80. Prenášaný kód vstupuje cez premennú MPRIS.

Podprogram SPRINT pridá na koniec reťazca B$ znak CR a takto upravený reťazec vyšle na tlačiareň prostredníctvom podprogramu SOUT.

```
70              $PRINT
71 '*************************
72 'PODPROGRAM    PRINT
73 '*************************
74 'NAHRADI PRIKAZ: PRINT#2 B$,
75 'PRICOM PRIPOJI ZNAK "CR" NA
76 'KONIEC RETAZCA B$
77 'PREDPOKLADAME, ZE TLACIARNI
78 '"PRT:" PRISLUCHA KANAL C.2.
79 B$=B$+CHR$(&0D)
80 FOR IPRI=1 TO LEN(B$)
90   MPRI$=MID$(B$,IPRI,1)
100  GOSUB $OUT
110 NEXT
120             RETURN
130             $OUT
131 '*************************
132 'PODPROGRAM - OUT
133 '*************************
134 'VYSLE JEDEN ZNAK, KTORY  JE
135 'ULOZENY  V  PREMENNEJ MPRI$
136 'NA TLACIAREN.
137 'PRENOS SA PREVEDIE CEZ ROZ-
138 'HRANIE CENTRONICS.
140 OUT 64,ASCII(MPRI$)
150 OUT 80,0:REPEAT:UNTIL(INP(80)AND 2)>
0
160 OUT 80,1
170             RETURN
```

Výpis 2. Riadenie tlačiarne v prípade nedokonalého prepojenia konektorov (905–V2)

a	161	A1	b	154	9A	c	159	9F	d	156	9C
e	146	92	f	170	AA	g	151	97	h	152	98
i	166	A6	j	175	AF	k	169	A9	l	184	B8
m	179	B3	n	176	B0	o	183	B7	p	158	9E
q	160	A0	r	157	9D	s	164	A4	t	150	96
u	165	A5	v	171	AB	w	163	A3	x	155	9B
y	189	BD	z	162	A2						

Tab. 1. Kódy malých písmen (905–T1)

```
5 LEN 64
10 OPEN "PRT:" AS#2
20 DIM PREV(26):'TABULKA KODOV MALYCH
25 'PISMEN
30 FOR I=1 TO 26:READ PREV(I):NEXT
40 DATA 161,154,159,156,146,170,151,152,
166,175,169,184,179,176,183,158,160,157,
164,150,165,171,163,155,189,162
50 POKE &705A,0:REM NEVYSIELAT KOD"LF"
60 'NA KONCI RIADKU
70 INPUT "SPRAVA";A$:B$=""
80 FOR I=1 TO LEN(A$):M=ASCII(MID$(A$,I,
1))
90   IF M>96 AND M<123 THEN M=PREV(M-96)
100  'PREVOD MALYCH PISMEN Z ASCII DO
      KODU SHARP
110  B$=B$+CHR$(M)
120 NEXT
130 PRINT#2;B$
140 GOTO 70
```

Výpis 1. Program na výpis textu zloženého z malych aj velkych písmen (905–V1)

EXPERTÍK

Ing. Jiří Mitlöhner, *Zahradní 228, Sázava, PSČ 285 06*

Účel programu:

1. **Příklad využití programovacího jazyka LOGO v oblasti zpracování seznamů a textů.**
2. **Předvádí a učí základní činnosti s databázovými systémy.**
3. **Umožňuje vytvořit jednoduché znalostní a výukové systémy.**
4. **Seznamuje s problematikou tvorby a využití jednoduché báze znalosti.**

Spuštění programu

Po vložení diskety zapíšeme LOGO a tím zavedeme do operační paměti interpret jazyka LOGO.

Na obrazovce vypsané texty echa interpreta odpovídají

a) 40 znakům v řádku obrazovky, pak zapíšeme LOAD „EXPERT1",
b) 80 znakům v řádku obrazovky, pak zapíšeme LOAD „EXPERT".

Dále postupujeme podle popisu funkce programu a nápovědy na obrazovce.

Popis báze znalostí

Báze znalostí je reprezentována stromovou datovou strukturou, kde každý uzel je označen kódem (kód odstavce libovolná znaková kombinace), popsán textem (text odstavce o max. délce 128 znaků) a je spojen s jedním nebo více následnými uzly odkazem na jejich kód (odkaz). Každé spojení může být popsáno textem (text odkazu). Stromová struktura báze znalostí může být téměř libovolně větvená i se zpětným provázáním. Programově je každý uzel báze znalostí reprezentován složeným seznamem, kód uzlu je jménem seznamu. Kódy uzlu jsou uloženy v seznamu (seznam jmen seznamů) se jménem „jména".

Počáteční uzel, to znamená odstavec, u kterého začíná funkce experta, musí být označen kódem 1. Funkce experta je ukončena odvoláním na kód (odkaz) KONEC, který nemusí být v bázi znalostí definován. Pořadí jednotlivých uzlů (odstavců) v bázi znalostí není rozhodující. Struktura je dána spojením jednotlivých uzlů odkazem na jejich kód.

Bázi znalostí můžeme uložit na disketu do souboru se zadaným jménem. Na disketě je ke jménu souboru automaticky připojena přípona .DAT, kterou při zadávání jména neuvádíme!

Výpis zkušební báze znalostí, stejně jako báze znalostí příkladu „Expertiza při volbě počítače a jeho programového vybavení" jsou uvedeny ve výpisu.

Popis funkce programu

Program Expertík umožňuje základní operace pro tvorbu, uložení, výpis a realizaci jednoduché báze znalostí. Základní funkce programu jsou nabídnuty po spuštění programu a v menu obsluhy. Během chodu programu jsou průběžně uváděny nápovědy, které umožňují využít všech jeho funkcí bez podrobného návodu.

Jednotlivé funkce Expertíka nabízené v menu obsluhy jsou následující:

E – expertík startuje. Start expertního nebo výukového programu na základě zadané báze znalostí.
Z – zápis baze znalosti. Zapsání nové, případně doplnění stávající báze znalostí z klávesnice. Způsob zadání je uveden po vyvolání funkce na obrazovce a je patrný ze zkušební báze a z příkladu báze znalostí.
V – vypis baze znalosti. Umožňuje výpis jednotlivých uzlů báze znalostí na obrazovce, případně jejich vymazání z databáze.
U – uloz bazi a disketu. Uloží bázi znalostí z operační paměti na disketu do souboru se zadaným jménem.
N – nahraj bazi z diskety. Přehraje z diskety do operační paměti bázi znalostí uloženou v souboru se zadaným jménem.
T – tiskne bazi na tiskarne. Vytiskne bázi znalostí z operační paměti na tiskárně, výpis není stránkován.
M – maz bazi znalosti. Vymaže celou bázi znalostí z operační paměti.
D – výpis souboru diskety. Vypíše jména všech souborů diskety.
K – konec programu. Ukončí práci programu Expertík s návratem do interpretu jazyka LOGO.

Popis programu

Program může sloužit jako učebnice jazyka LOGO pro zpracování seznamů. Jednotlivé procedury a funkce je možné po zadání vstupních parametrů volat samostatně a sledovat tak jejich funkci. Proto jsou dále vyjmenovány všechny procedury a funkce, udán popis jejich činnosti a uvedeny případné vstupní paramatry.

Základní procedury

START Spuštění programu. Definuje počet znaků jedné řádky obrazovky.
Volá: POPIS, EXSYS

POPIS Vypíše úvodní text s dotazem na nahrání báze znalostí z diskety.
Volá: ZDISKU

EXSYS Realizuje výpis hlavního menu obsluhy a volbu jednotlivých funkcí programu.
Volá: MENU, VOLBA

MENU Vypíše text menu obsluhy na obrazovku.
Volá: –

VOLBA Zajišťuje volbu funkce programu podle zadání obsluhy z klávesnice.
Volá: ZAPIS, EXPERT, VYPIS, KONEC, NADISK, ZDISKU, MAZE, TISKNE, DISKETA.

EXPERT Provádí výpis jednotlivých uzlů báze znalostí podle odpovědi obsluhy na texty spojů (odkazů), realizuje mechanismus vlastního expertního systému.
Volá: STA, CHYBA

ZAPIS Realizuje zápis báze znalostí z klávesnice do operační paměti.
Volá: ZZ

VYPIS Provádí výpis báze znalostí po jednotlivých uzlech na obrazovku, a případně realizuje zadané mazání zobrazeného uzlu.
Volá: VYPS

NADISK Kopíruje bázi znalostí z operační paměti do pojmenovaného souboru na disketě. Ptá se na jméno souboru, po zadání kontroluje jeho existenci. Existuje-li soubor, čeká na odpověď k přemazání stávajícího souboru.
Volá: WTE

ZDISKU Načte obsah zadaného souboru báze znalostí z diskety do operační paměti. Testuje existenci souboru a dává obsluze příslušné nápovědy.
Volá: RED

MAZE Po dotazu provede vymazání celé báze znalostí v operační paměti.
Volá: MAZ

TISKNE Vypíše bázi znalostí z operační paměti na tiskárnu. Dává obsluze příslušné nápovědy.
Volá: OUT

DISKETA Zobrazí na obrazovce seznam souborů na disketě.
Volá: –

Pomocné procedury a funkce

STA Zajišťuje mechanismus výpisu expertního systému až po nalezení kódu uzlu KONEC.
Vstup: jm kód prvního uzlu
Volá: VYBER

VYBER Podle kódu najde uzel, který vypíše na obrazovku. Načte

z klávesnice kód spoje k dalšímu uzlu.

Vstup: p kód uzlu
Volá: PIS, ITM

PIS Vypište číslo a texty spojů (odstavců) nalezeného uzlu.

Vstup: i pořadové číslo spoje v uzlu (objektu seznamu)
pr jméno uzlu (seznamu)
Volá: ODSTAVEC

CHYBA Výpis echa chyby báze znalostí.

ZZ Realizuje zápis jednoho seznamu (uzlu báze znalostí) z klávesnice nebo z diskety (záleží na výběru vstupního zařízení) až po načtení kódu, ukončujícího zápis.

Vstup: kod kód ukončující zápis
Vrací: true byl ukončen zápis
Volá: VSTUP

VSTUP Zařadí seznam z klávesnice nebo souboru diskety do určeného složeného seznamu.

VYPS Vypíše jeden uzel na obrazovku a testuje stisk řídících kláves, podle kterých zruší další výpis, pokračuje výpisem dalšího uzlu, nebo volá proceduru mazající zobrazený uzel (výpis seznamu).

— *Vstup:* jm kód zpracovávaného uzlu (jméno seznamu)
Volá: TISK, VYMAZ

TISK Je-li v operační paměti umístěna báze znalostí, vypíše kódem zadaný uzel na obrazovku (vypíše seznam).

Volá: TISKNI

TISKNI Zobrazí na obrazovce jeden uzel (zobrazí objekty seznamu).

Vstup: jm kód uzlu (jméno seznamu)
Volá: RADEK

RADEK Vypíše na obrazovku všechny objekty zadaného seznamu.

Vstup: r jméno seznamu

VYMAZ Vymaže z operační paměti zadaný seznam (uzel) a zároveň vymaže i jeho jméno ze seznamu kódů uzlů (jmen seznamů).

Volá: DEL

DEL Vymaže ze zadaného seznamu označený objekt.

Vstup: p objekt
js jméno seznamu
Volá: VYPUST, NAJDI

VYPUST Vymaže ze zadaného seznamu objekt se zadaným pořadovým číslem.

Vstup: c pořadové číslo objektu
s jméno seznamu
Volá: ROT

ROT Vrací o zadané číslo rotovaný seznam.

Vstup: c počet rotací o jeden objekt
s jméno seznamu

NAJDI Testuje přítomnost zadaného objektu v seznamu a vrací jeho pořadové číslo v seznamu.

Vstup: p objekt
s jméno seznamu
Volá: NAJ

NAJ Vrací pořadové číslo objektu v zadaném seznamu.

Vstup: c pořadové číslo prvku od kterého probíhá hledání
p objekt
s jméno seznamu

WTE Zapíše do souboru na disku seznamy, jejichž jména jsou zadaná v seznamu jmen (uzly báze znalostí).

Vstup: sb jméno souboru na disketě
jm jméno seznamu jmen
Volá: OUT

OUT Zapíše na zvolené zařízení seznamy, jejichž jména jsou zadaná v seznamu jmen.

Vstup: j jméno seznamu jmen
Volá: OUTS

OUTS Zapíše na zvolené zařízení objekty seznamu (spoje uzlu).

Vstup: s jméno seznamu

RED Čte ze souboru na disketě seznamy, jejichž jména jsou zadána v seznamu jmen (uzle báze znalostí).

Vstup: sb jméno souboru na disketě
Volá: ZAP

ZAP Čte ze zvoleného vstupního zařízení seznamy až do nalezení značky EOE (seznamu s textem EOE).

Volá: ZZ

MAZ Provede vymazání všech proměnných v operační paměti a inicializuje prázdný seznam jmen, vypíše hlášení.

ITM Vrací objekt se zadaným pořadovým číslem ze zadaného seznamu, při zadání většího čísla než je poslední objekt vrací poslední objekt seznamu.

Vstup: c pořadové číslo objektu v seznamu
s jméno seznamu

RP Provede výpis zvoleného počtu zadaného znaku na zvolené místo obrazovky.

Vstup: c souřadnice výpisu na obrazovce
p počet vypisovaných znaků
z vypisovaný znak

SP Výpis zadaného textu na zvolené místo obrazovky

Vstup: c souřadnice výpisu na obrazovce
t jméno seznamu
s vypisovaným textem

ODSTAVEC Provádí výpis textu ze seznamu do odstavce zadaného souřadnicemi na obrazovce.

Vstup: poz souřadnice na obrazovce
seznam jméno seznamu
Volá: LINE

LINE Vypíše jeden text seznamu do jednoho řádku odstavce.

Volá: KONRADKU

KONRADKU Hlídá fyzický konec řádku na obrazovce.

SAVEC Pomocná procedura pro zápis zdrojového programu s echem na přepsání existujícího souboru na disku.

Vstup: spec jméno souboru

Výpis 1. *Program Expertík* (931-V1)

```
(931-V1)

TO START
POPIS
EXSYS
END

TO POPIS
TS CT RP [2 8] 19 [*]
SP [4 8] [* E X P E R T I K *]
RP [6 8] 19 [*]
SETCURSOR [8 4]
(TYPE "( "C ") CHAR 32 [Ing.J.Mitlohner 1988])
SP [10 2] [Program umoznuje tvorbu a spusteni]
SP [11 2] [jednoduchého expertního systemu.]
SP [12 2] [Funkce EXPERTIKA je dana bazi]
SP [13 2] [znalosti, kterou muzete vytvorit]
SP [14 2] [a zaznamenat na disk. Vytvorenou]
SP [15 2] [bazi je mozne vyuzit pri novem]
SP [16 2] [startu EXPERTIKA.]
RP [19 2] 36 [_]
SP [21 2] [Chces nahrat ZNALOSTI z diskety ?]
SP [22 30] [Ano / Ne]
IF RC = "A [ZDISKU]
END

TO EXSYS
MAKE "k "FALSE
MENU
VOLBA
IF :k [CT SP [2 8] [PROGRAM EXPERTIK JE UKONCEN !]
    WAIT 40 CT SETCURSOR [1 1] STOP]
EXSYS
END

TO MENU
TS CT
```

```
SP [2 5] [M E N U - E X P E R T I K A]
RP [3 0] 38 [_]
SP [5 5] [E - expertik startuje]
SP [7 5] [Z - zapis baze znalosti]
SP [8 5] [V - vypis baze znalosti]
SP [15 5] [K - konec programu]
SP [9 5] [U - uloz bazi na disketu]
SP [10 5] [N - nahraj bazi z diskety]
SP [11 5] [M - maz bazi znalosti]
SP [12 5] [T - tiskni bazi na tiskarne]
SP [13 5] [D - vypis souboru diskety]
RP [19 0] 38 [_]
SP [21 3] [Stiskni vyznacenou klavesu !]
END

TO VOLBA
MAKE "z RC
IF :z = "Z [ZAPIS]
IF :z = "E [EXPERT]
IF :z = "V [VYPIS]
IF :z = "K [KONEC]
IF :z = "U [NADISK]
IF :z = "N [ZDISKU]
IF :z = "M [MAZE]
IF :z = "T [TISKNE]
IF :z = "D [DISKETA]
END

TO EXPERT
CT
IF NOT NAMEP "jmena [MAKE "jmena []]
IF NOT NAMEP "1 [CHYBA] [MAKE "jm 1 STA :jm]
END

TO ZAPIS
CT PR [ZAPIS BAZI ZNALOSTI VE TVARU :]
RP [1 0] 36 [_] PR []
PR [Kod odstavce]
PR [Nazev odstavce]
PR [[TEXT ODKAZU 1] ODKAZ 1]
PR [[TEXT ODKAZU 2] ODKAZ 2]
PR [a t d dalsi odkazy]
PR [ENTER]
RP [8 0] 36 [_]
RP [22 0] 36 [_]
SP [23 2] [ZAPIS UKONCI DVOJIM STISKEM ENTER]
IF NOT NAMEP "jmena [MAKE "jmena []]
SETCURSOR [9 1]
IF ZZ [] [STOP]
ZAPIS
END

TO VYPIS
IF NOT NAMEP "jmena [MAKE "jmena []]
VYPS :jmena
END

TO NADISK
CT SP [8 2] [ZAPIS JMENO BAZE ZNALOSTI]
SETCURSOR [10 2] TYPE [NA DISKETE ! -]
MAKE "soubor RW
IF EMPTYP :soubor [STOP]
MAKE "soubor WORD :soubor ".DAT
IF FILEP :soubor [SP [12 2] [BAZE NA DISKETE
        EXISTUJE PREMAZ Ano / Ne]
      IF RC = "A [ERASEFILE :soubor] [STOP]]
SP [14 2] [PROBIHA ZAPIS NA DISKETU !]
WTE :soubor :jmena
END


TO ZDISKU
CT SP [8 2] [ZAPIS JMENO BAZE ZNALOSTI]
SETCURSOR [10 2] TYPE [NA DISKETE ! -]
MAKE "soubor RW
IF EMPTYP :soubor [STOP]
MAKE "soubor WORD :soubor ".dat
IF NOT FILEP :soubor [SETCURSOR [12 2]
    (TYPE [BAZE -] :soubor [- NEEXISTUJE !])
     WAIT 30 STOP]
SP [14 2] [PROBIHA CTENI DAT Z DISKETY !]
RED :soubor
END

TO MAZE
CT SP [8 5] [POZOR ! CHCES OPRAVDU VYMAZAT]
SP [10 5] [VSECHNA DATA ZNALOSTI ? Ano / Ne]
IF RC = "A [MAZ]
END

TO TISKNE
CT SP [8 2] [ZAPNI A PRIPRAV TISKARNU !]
SP [10 2] [PO TE STISKNI NEKTEROU KLAVESU !]
DRIBBLE "LPT1
MAKE ":z RC
SP [14 2] [PROBIHA TISK BAZE ZNALOSTI !]
OUT :jmena
NODRIBBLE
END

TO DISKETA
CT SP [2 2] [VYPIS SOUBORU Z DISKETY]
SETCURSOR [4 0]
DIR
SP [24 2] [UKONCI STISKEM NEKTERE KLAVESY !]
MAKE :z RC
END

TO STA :jm
IF EQUALP :jm "KONEC [STOP]
MAKE "jm VYBER :jm
STA :jm
END

TO VYBER :p
MAKE "pr THING :p
CT PR [] ODSTAVEC CURSOR FIRST :pr
PIS 1 BF :pr
PR [] PR []
REPEAT :sirka [TYPE "-] PR []
TYPE [Zapis cislo zvoleneho odstavce !]
MAKE "z RC
IF :z = CHAR 27 [OP 1]
IF OR NOT NUMBERP :z :z = 0 [MAKE "z 1]
OP LAST ITM (1 + :z) :pr
END

TO PIS :i :pr
IF EMPTYP BF :pr [STOP]
PR [] PR []
(TYPE "( :i ") "-)
ODSTAVEC CURSOR FIRST FIRST :pr
PIS :i + 1 BF :pr
END

TO CHYBA
SP [2 10] [ZNALOSTNI BAZE NENI UPLNA !]
WAIT 100
END


TO ZZ :kod
MAKE "jm RL
IF EQUALP :jm :kod [OP "TRUE]
MAKE "jm FIRST :jm
MAKE :jm []
MAKE "s []
MAKE "jmena FPUT :jm :jmena
VSTUP
REPEAT COUNT :s [MAKE :jm FPUT FIRST :s
                 THING :jm MAKE "s BF :s]
OP "FALSE
END


TO VSTUP
MAKE "ss RL
MAKE "s FPUT :ss :s
IF NOT EMPTYP :ss [VSTUP]
END

TO VYPS :jm
CT SETCURSOR [2 1] TISK
SP [21 2] [Pokracuj stiskem mezerniku !]
SP [22 2] [Ukonci stiskem ESC]
SP [23 2] [Vymaz stiskem Ctrl C]
MAKE "z RC
IF :z = CHAR 3 [VYMAZ]
IF :z = CHAR 27 [STOP]
IF EMPTYP :jm [STOP]
VYPS ROT 1 :jm
END

TO TISK
IF EMPTYP :jm [PR [CHYBI DATA ZNALOSTI !]
               STOP] [TISKNI :jm]
END

TO TISKNI :jm
MAKE "prom FIRST :jm
PR :prom
MAKE "rad THING FIRST :jm
RADEK :rad
END

TO RADEK :r
IF EMPTYP :r [STOP]
PR FIRST :r
RADEK BF :r
END

TO VYMAZ
ERN THING "prom
DEL :prom "jmena
MAKE "jm :jmena
END

TO DEL :p :js
MAKE "s THING :js
MAKE :js VYPUST NAJDI :p :s :s
END

TO VYPUST :c :s
MAKE "c :c - 1
OP ROT :c BF ROT (COUNT :s) - :c :s
END

TO ROT :c :s
REPEAT :c [MAKE "s BL FPUT LAST :s :s]
OP :s
END

TO NAJDI :p :s
IF NOT MEMBERP :p :s [OP 0]
OP NAJ 1 :p :s
END

TO NAJ :c :p :s
IF NOT EQUALP :p FIRST :s [MAKE "c NAJ :c
                                + 1 :p BF :s]
OP :c
END


TO WTE :sb :jm
OPEN :sb
SETWRITE :sb
OUT :jm
SHOW [EOE]
SHOW [EOE]
SETWRITE []
CLOSE :sb
END


TO OUT :j
IF EMPTYP :j [STOP]
MAKE "js FIRST :j
PR :js
OUTS THING :js
OUT BF :j
END

TO OUTS :s
IF EMPTYP :s [STOP]
```

```
PR FIRST :s
OUTS BF :s
END

TO RED :sb
MAKE "jmena []
OPEN :sb
SETREAD :sb
ZAP
CLOSE :sb
SETREAD []
END

TO ZAP
IF ZZ [[EOE]] [STOP]
ZAP
END

TO MAZ
ERNS
MAKE "jmena [] MAKE "k "FALSE
MAKE "z "X

CT SP [8 5] [VSECHA DATA ZNALOSTI BYLA
            VYMAZANA !]
WAIT 30
END

TO ITM :c :s
MAKE "cou COUNT :s
MAKE "cou :cou - 1
IF :c > :cou [MAKE "c :cou]
IF :c < 1 [MAKE "c 1]
OP ITEM :c :s
END

TO RP :c :p :z
SETCURSOR :c REPEAT :p [TYPE :z]
END

TO SP :c :t
SETCURSOR :c PR :t
END

TO ODSTAVEC :poz :seznam
SETCURSOR :poz

MAKE "cisrad FIRST CURSOR
MAKE "ppoz LAST :poz
REPEAT COUNT :seznam [LINE]
END

TO LINE
IF KONRADKU [SETCURSOR LIST ((FIRST CURSOR)
            + 1) :ppoz]
(TYPE FIRST :seznam CHAR 32)
MAKE "seznam BF :seznam
END

TO KONRADKU
OP (((LAST CURSOR) + (COUNT
        FIRST :seznam)) > :sirka)
END

TO SAVEC :spec
IF FILEP WORD :spec ".LF [PR [SOUBOR EXISTUJE,
                    PREHREJ A / N] IF RC = "A
                    [ERASEFILE :spec] [STOP]]
SAVE :spec
END
```

Výpis 2. Báze znalostí (931-V2)

```
230
Uz jste nekdy zkousel programovat?
[Jiste, jsem profesional] 220B
[Na necem se to prece musim naucit!] 221
[Nekolik jednoduchych programu jsem vypotil] 221

220B
Jestli - jste si dostatecne overil sve programatorske
dovednosti, pak si urcite vyberet sam, podle svych
financnich moznosti
[Pocitac kompatibilni s IBM PC XT. Cela rada vyrobcu,
rozsahle programove vybaveni. Zbyva sehnat software.] 31

221
Zkuste se nejdrive seznamit s programovanim na levnejsim
pocitaci v klubech nebo u svych znamych, teprve potom se
rozhodnete!
[Pokracuj stiskem nektere klavesy] 17

220AA
Cela rada domacich 16 bitovych pocitacu. Pred koupi nutno
vlastnosti, cenu, moznosti nakupu konzultovat ( literatura,
kluby )
[I pres konzultace si na nakup netroufam] 221
[Pro zacatek zvolim neco levnejsiho] 210
[Dekuji, uz jsem si vybral] KONEC

220A
Ve svete velky vyber vetsinou 16 bitovych pocitacu. Pro nase
podminky nejlepe
[ATARI ST] 220AA
[COMMODORE AMIGA] 220AB

220
Jestli jste jeste nikdy neprogramoval, doporucuji nizsi
cenovou kategorii. Jinak zalezi na tom, jak chcete pocitac
vyuzivat.
[Amatersky pro zabavu a pouceni v domacnosti] 220A
[Profesionalne pro zhotoveni a prodej programoveho vybaveni]
220B

210
K dale nabizenym pocitacum si muzes prikoupit disketovou
jednotku. Uzitne vlastnosti pocitace se mnohonasobne zvysi!
[Pokracuj stiskem nektere klavesy] 205

205A
Ve svete jeden z nejrozsirenejsich a nejlepsich ve sve
kategorii. U nas dostupny pres inseraty. Malo rozsireny.
[Vyber mi vyhovuje] 202
[Malo informaci] 203
[Chci jiny pocitac] 204

205
Bezny zajem - bezne zbozi
[COMMODORE 64] 205A
[Jine pocitace] 201

204
Chcete obetovat do pocitace vice penez?
[Ano] 210
[Ne] 201

203
Prostudujte si literaturu, nebo se poradte v pocitacovych
klubech! Vhodne casopisy - ELEKTRONIKA, AMATERSKE RADIO,
VTM.
[Pokracujte stiskem nektere klavesy] KONEC

202
Mnoho uspechu pri koupi pocitace! Chcete poradit pri volbe
programoveho vybaveni?
[Ano] 10
[Ne] KONEC

201D
U nas pomerne malo rozsireny. Dostupny na inzerat. Omezene
programove vybaveni.Dobra grafika a standartni klavesnice.
[Vyber mi vyhovuje] 202
[Malo informaci] 203
[Chci jiny pocitac] 204

201C
U nas rozsireny pocitac ( TUZEX, maloobchod, inserce )
[Vyber mi vyhovuje] 202
[Malo informaci] 203
[Chci jiny pocitac] 204

201B
U nas nejrozsirenejsi pocitac. Nevyrabi se. Nestandartni
klavesnice, bohate programove vybaveni. Dostupny na
inseraty.
[Vyber vyhovyje] 202
[Malo informaci] 203
[Chci jiny pocitac] 204

201A
Tuzemsky vyrobek, bezne dosazitelny v obchodech. Temer
ekvivalent SINCLAIR SPECTRUM. Rozsahle programove vybaveni.
[Jsem spokojeny s vyberem] 202
[Malo informaci] 203
[Chci jiny pocitac] 204
```

201
Nejcastejsi pripad, ale za malo penez malo muziky!
[DIDAKTIK GAMA] 201A
[SINCLAIR SPECTRUM] 201B
[ATARI 800 XL / 130 XE] 201C
[SHARP MZ - 800] 201D

200
Nesnadna odpoved! Kolik chces investovat penez?
[Co nejmene] 201
[Do 10 tis.Kcs nebo 250 DM] 205
[Do 25 tis.Kcs NEBO 500 DM] 210
[Do 90 tis.Kcs nebo 1000 DM] 220
[Nezalezi na penezich] 230

30
Nevzdavejte se tak brzy! I v jinych jazycich musite umet formulovat ulohu, premysleni a drina Vas stejne nemine!
[Radeji uz toho necham] 19
[Jeste to zkusim] 52

52
Spravny zacatek Vasi programatorske kariery, ted jde o to jak jste si pri tom vedl!
[Pokracuj stiskem nektere klavesy] 17B

17B
S jakymi vysledky jsi uspel v programovani?
[Dokazi zapsat jednoduche programy, mam zajem o programovani v jinem jazyce] 30
[Samostatne napisi a odladim i slozite programy vyuzivajici zapisu a vypisu dat na disketu (kazetu)] 31
[Pres vsechnu snahu mi dela potize sestaveni i jednoduchych programu] 19

17A
LOGO je rozsirene na vetsine typech pocitacu. Obrat se na pocitacovy klub,stanice mladych techniku a pod. zarizeni!
[Pokracuj stiskem nektere klavesy!] 17

17
Pro vstup do sveta programovani doporucuji jazyk LOGO. Je moderni a vytvorite si dobry zaklad pro vyuku dalsich jazyku.
[Nemohu LOGO na svuj pocitac sehnat] 17A
[Seznamil jsem se s jazykem LOGO] 17B

51
Nedostal jste se sice daleko, ale jste na spravne ceste
[Pokracuj stiskem nektere klavesy!] 17

50
Nezvolil jste dobry vstup do zivota programatora! Radeji skuste rychle jeste neco jineho!
[Pokracuj stiskem nektere klavesy] 17

20
Chcete mou radu, pak nejste s programovanim prilis daleko. V jakem jazyku jste prevazne programoval?
[BASIC] 50
[KAREL] 51
[LOGO] 52
[Jednim, nebo vice z PASCAL,PROLOG,C,LISP,ASSEMBLER,COBOL] 31
[V zadnem a nebo ve vsech, ale ne moc uspesne] 17

44
Krome specializovanych prostredku jazyky PROLOG, LISP, MAMPS, C
[Ukonci stiskem nektere klavesy] KONEC

43
Cela rada specialnich programovych prostredku na pr. AUTOCAD,CHART,DrHALLO,ORDCAD a jine. Doplnit jazyky C, LISP, pr. PASCAL.
[Ukonci stiskem nektere klavesy!] KONEC

42
Integrovany software na pr. dBASE,FRAMEWORK,SUPERCALC a jine. Z jazyku pak C - jazyk, PASCAL
[Ukonci stiskem nektere klavesy!] KONEC

41
Existuje bohata nabidka specielniho vybaveni na pr.STATGRAPHICS, EUREKA a dalsi. Jinak jazyky FORTRAN, PASCAL, BASIC
[Ukonci stiskem nektre klavesy!] KONEC

40
Drive bych doporucil ASSEMBLER. Ten je vsak zavisly na typu pocitace. Nyni jednoznacne doporucuji C - jazyk !
[Ukonci stiskem nektere klavesy!] KONEC

31
Mate talent pro programovani! Dale zalezi na oboru, ve kterem tento talent chcete uplatnit
[Rizeni stroju, robotu, technologii] 40
[Vedecko technicke vypocty, matematika] 41
[Ekonomicke vypocty, administrativa] 42
[Projektovani, pocitacova grafika] 43
[Tvorba zakladniho programového vybaveni] 40
[Expertni systemy, znalostni systemy, umela inteligence] 44

19
Programator z Vas nebude, ted uz nemuzete nic pokazit, zkuste jazyk BASIC, mozna ze Vas uspokoji!
[Ukonci stiskem nektere klavesy] KONEC

21
Pokuste se sehnat programovaci jazyk LOGO, ale nenechte se na zacatku odradit neuspechem! Vytrvejte!
[Ukonci stiskem nektere klavesy] KONEC

18
Zkuste nejdrive programovaci jazyk KAREL
[Programovani se Vam darilo a bavi Vas?] 17
[Hra s robotem me nezaujala, ale programovat bych chtel v jinem jazyce] 21
[Robot me nechce poslouchat,jeho slozitejsi cinnost nedokazi formulovat] 19
[Nesehnal jsem program KAREL na muj pocitac] 21

1
Mate problem s vyberem pocitace a programoveho vybaveni? Expertik Vam pomuze! Staci odpovidat na jeho dotazy a poradi Vam!
[Chci vyuzit sluzeb expertika pro vyber programoveho vybaveni] 10
[Chci si poridt svuj prvni pocitac] 200
[Nemam zajem o pomoc expertika] KONEC

10
Vyber zavisi na Vasich zajmech, dosavadnich znalostech a typu pocitace.
[Nevim jeste, jaky pocitac si poridim] 200
[Pocitac jsem si uz vybral, nemam zadne zkusenosti s obsluhou a programovanim] 15
[Mam pocitac a znam zaklady programovani, chci se programovani systematicky venovat] 20

15
Musite se rozhodnout k jakym ucelum chcete Vas pocitac prevazne vyuzivat.
[Hry a nebo hotove, jednoucelove programy] 16
[Mam zajem se naucit programovat] 17
[Jeste nevim, chci se rozhodnot pozdeji] 18

16
Hotove programy a hry Vam nabidnou pocitacove kluby, stanice mladych techniku, domy mladeze a podobne.
[O jine programy uz nemam zajem, nebo az pozdeji] KONEC
[Prece jenom bych si rad zkusil neco sam naprogramovat] 18

[EOE]
[EOE]

INTERFEJS TISKÁRNA – ATARI

Ing. J. Kodera, *Hůrka 1058, 278 01 Kralupy n. V.*

Možnost připojení tiskárny je jednou ze základních podmínek pro skutečně efektivní využití mikropočítačů všech typů. U nás stále rozšířenější osmibitové domácí mikropočítače ATARI umožňují připojení různých periferních zařízení (disketová jednotka, tiskárna, datový magnetofon), přes sériový port. Tento port však bohužel neodpovídá žádnému běžnému standardu a tak lze bez problémů připojit pouze periférie dodávané přímo firmou nebo vybavené příslušným interfejsem (z tiskáren např. ATARI 1027 nebo 1029, SEIKOSHA GP500AT). *(Zapojení bylo vyvinuto a zasláno redakci v roce 1987 – pozn. red.)*

Běžnou tiskárnu vybavenou např. rozhraním Centronics lze v zásadě připojit dvěma způsoby:

1. Využít paralelní port počítače používaný jinak pro připojení ovladačů (joysticků), obsluhovaný v počítači obvodem PIA 6520, což znamená napsat příslušný obslužný program pro počítač. Nevýhodou tohoto řešení je, že není podporováno žádným z běžného programového vybavení.

2. Zkonstruovat interfejs, který se připojí mezi sériový port a tiskárnu. Toto řešení je sice náročnější, zato však daleko pružnější a zachovává všechny výhody počítače včetně podpory operačního systému, BASICu a veškerého softwaru (např. textových editorů),

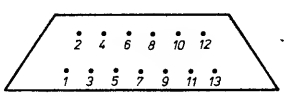
Druhé řešení je obsahem následujícího příspěvku.

Princip komunikace počítače s periferními zařízeními

Veškeré periférie se k počítači připojují přes třináctipólový konektor, umístěný na zadní straně počítače. Popis konektoru je na **obr. 1.** Pro připojení interfejsu se využívají pouze vývody č. 3, 4, 5, 7, 10. Periférie (s výjimkou magnetofonu) jsou vybaveny vlastní „inteligencí" a jsou připojeny přes budiče s otevřenými kolektory, takže může být paralelně připojeno více zařízení najednou, aniž by se vzájemně ovlivňovala. Přesto jsou standardní periférie vybavovány dvěma konektory podle **obr. 1** propojenými paralelně.

Všechny napěťové úrovně jsou TTL, logika je pozitivní.

Komunikace je sériová asynchronní, značky jsou osmibitové s jedním START bitem (log.0) a jedním STOP bitem (log.1).



Obr. 1. Konektor počítačů Atari pro sériový styk s periférierni (901–1). *Popis vývodů:*

1	**CLOCK IN**	nepoužívá se
2	**CLOCK OUT**	nepoužívá se
3	**DATA IN**	sériová vstupní datová linka
4	**GND**	zem
5	**DATA OUT**	sériová výstupní datová linka
6	**GND**	zem
7	**CMDn**	označení povelu
8	**MOTOR CONTROL**	řízení motoru dat. magnetofonu
9	**PROCEED**	nepoužívá se
10	**+ 5 V**	napájecí napětí
11	**AUDIO IN**	vstup akustického signálu
12	**NC**	nepoužívá se
13	**INTERRUPT**	nepoužívá se

Rychlost je 19 200 Bd, používá se standardní soubor znaků ASCII, přičemž znaky zobrazované na stínítku inverzně mají nastavený nejvyšší bit do log.1. V klidu je na kontaktech (samozřejmě s výjimkou GND) úroveň log.1.

Jak již bylo řečeno, komunikace je „inteligentní". V praxi to vypadá např. po příkazu LIST „P:" následovně:

1. Počítač oznámí všem připojeným periferním zařízením logickou úrovní 0 na vývodu CMD, že vydává blok povelu. Zároveň vyšle pět povelových bajtů na vývod DATA OUT. Složení povelového bloku je: jeden bajt označení periférie (40H pro tiskárnu, „P:"), jeden bajt označení povelu (53H je žádost o status), dva bajty dodatečné informace (pro tiskárnu dvakrát 0), jeden bajt zabezpečení (součet všech čtyř předešlých bajtů s uvážením přenosu např. instrukcí ADC většiny mikroprocesorů).

2. Po skončení signálu CMD odpoví oslovená periférie přes vývod DATA IN podle druhu povelu. Odpověď na žádost o status je sedmibajtová.

3. Počítač vydá další povelový blok, opět zároveň se signálem CMD. Tento blok je stejný jako v bodě 1, pouze je jiné označení povelu (57H pro zápis dat na periferní zařízení s kontrolním součtem).

4. Oslovená periférie opět odpoví, tentokrát jen jedním bajtem (41H jako kladné potvrzení povelu).

5. Počítač vydá jeden blok dat, standardně dlouhý 40 datových bajtů plus jeden bajt zabezpečení (stejné jako u povelového bloku).

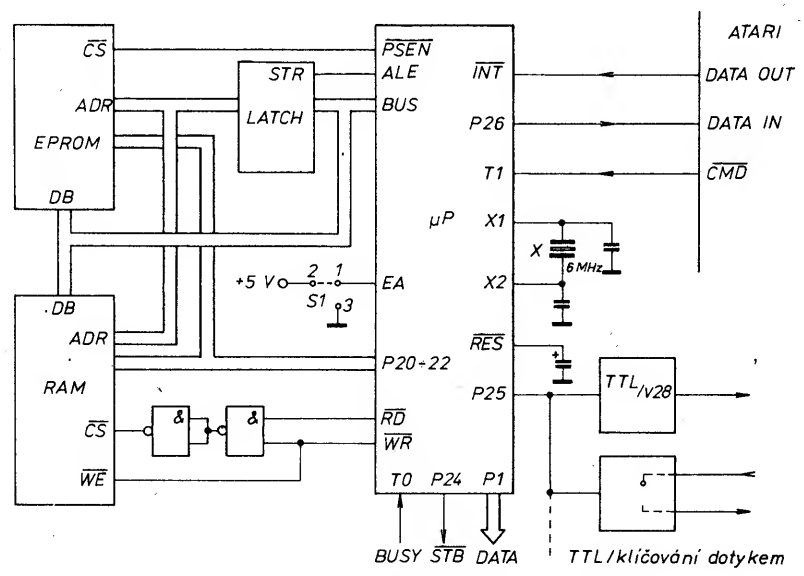
6. Periférie odpoví jedním bajtem (41H jako kladné potvrzení), zpracuje přijatá data a oznámí počítači, že komunikace může pokračovat (bajtem 43H).

7. Není-li komunikace ukončena, pokračuje počítač bodem **3.**

V průběhu komunikace může samozřejmě dojít k poruše; pokud počítač nedostane včas odpověď na svůj povel, opakuje ho (standardně celkem 13×), pokud ani jednou přitom nedostane odpověď, vyhlásí chybu č. 138 (TIME OUT). Stejně je tomu v případě, že počítač nedostane kladné potvrzení o přijetí dat.

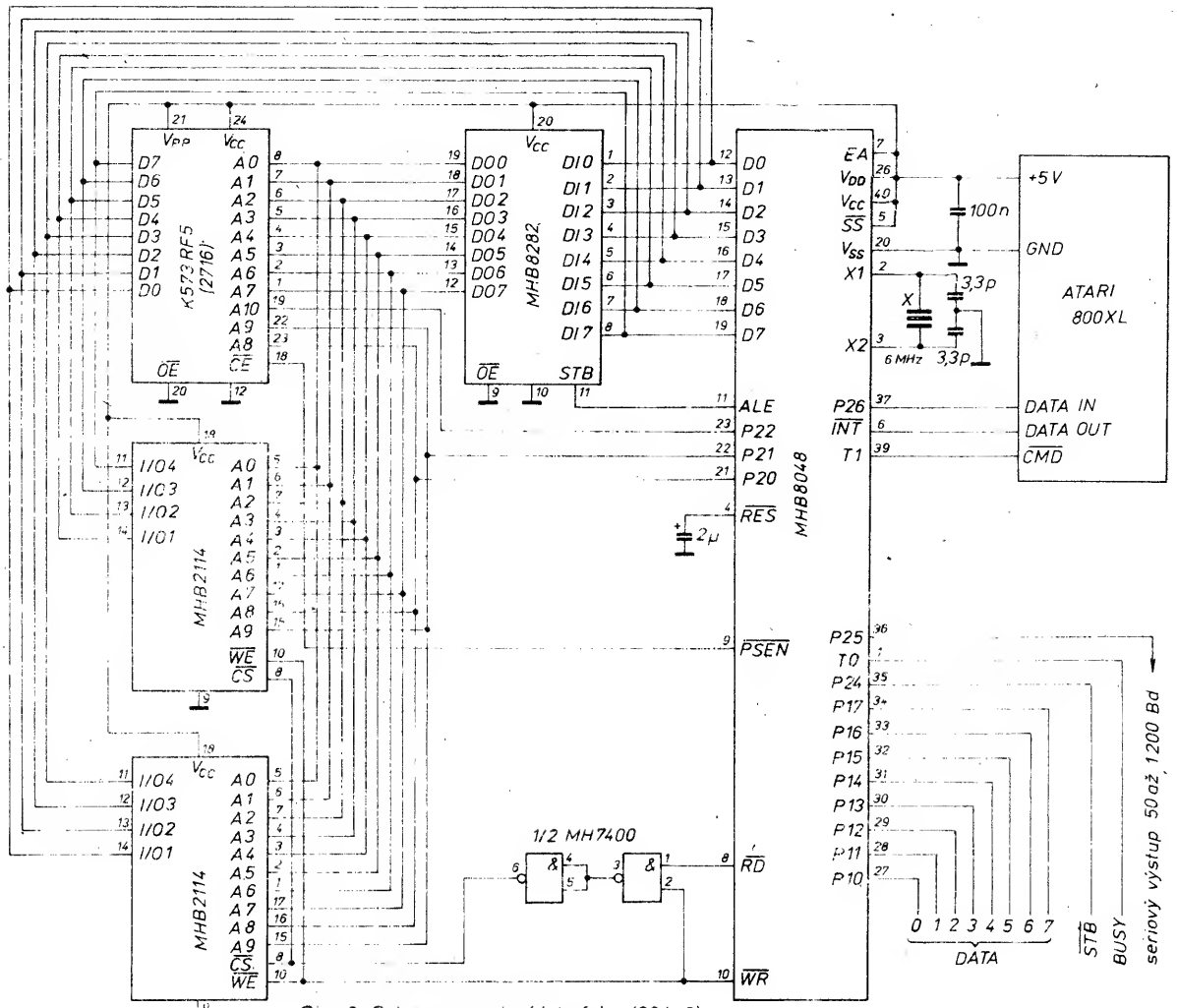
Realizace a stavba zařízení

Z uvedeného popisu způsobu komunikace počítače s periférierni vyplývá, že realizace interfejsu je možná prakticky jen při použití mikroprocesoru, který je řízen programem zajišťujícím všechny potřebné funkce a který zároveň umožňuje interfejs přizpůsobit prakticky libovolné tiskárně. Jakékoli čistě hardwarové řešení by vyšlo neúměrně složité a nákladné. Jako prakticky ideální řešení se nabízí použití některého jednočipového mikropočítače z řady 48, který ke své funkci potřebuje minimum dalších součástek a je relativně dostupný i v ČSSR. Principiální schéma interfejsu je na **obr. 2.** Pokud použijeme mikropočítač 8048 nebo 8035, je třeba připojit obě vnější paměti (RAM i EPROM). Výhodnější je použít mikropočítač 8747 se zabudovanou vnitřní pamětí EPROM, po-



Obr. 2. Blokové schéma zapojení interfejsu (901–2).

Propojka S1 je pro typ 8748 v poloze **1–3,** pro ostatní typy v poloze **1–2.**



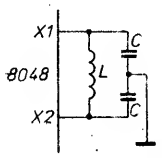
Obr. 3. Schema zapojení interfejsu (901–3)

tom odpadá vnější programovatelná paměť, avšak tento obvod je hůře dostupný a pro většinu amatérů je velmi obtížné zajistit naprogramování vnitřní paměti. Výhodné je i použití mikropočítače 8039, který má postačující kapacitu vnitřní RAM, s dostupností tohoto obvodu je to však ještě horší (k. p. Tesla zatím dodává typy 8048, 8035 a 8748).

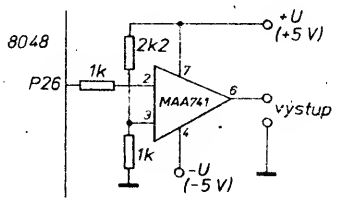
Na místě vnější RAM je možné použít libovolnou paměť RAM s kapacitou alespoň 64 bajtů, vyhoví např. dva tuzemské obvody MHB2114 s organizací 1k×4 bity, ze zahraničních typů např. 6116 (s kapacitou 2 kB) apod. Latch je osmibitový paralelní střádač, např. MHB8282, SN74373, apod. EPROM je libovolná s kapacitou alespoň 1 kB – např. MHB8708C, výhodnější z hlediska zachování jediného napájecího napětí je však typ 2716, resp. sovětský ekvivalent K573RF5.

Jedna z možností konkrétní realizace je na **obr. 3.** Jsou zde použity obvody, o kterých se domnívám, že by měly být nejsnáze dostupné. Jak však bylo řečeno, možností je mnoho a záleží na možnostech a na štěstí každého zájemce o stavbu interfejsu, které konkrétní obvody se mu podaří sehnat. Z tohoto důvodu rovněž neuvádím výkres desky s plošnými spoji; navíc se domnívám, že pro stavbu jednoho kusu je nejvýhodnější využít některou univerzální zapojovací desku a drátové propojení. Výhodné je však použít sokly pro mikroprocesor a samozřejmě pro EPROM.

Kritickým místem celé konstrukce je zřejmě krystal. Díky velmi vysoké přenosové rychlosti bylo nutno časování odvodit přímo z doby trvání instrukčního cyklu, která je pro krystal 6 MHz 2,5 μs. Odpovídající kmitočet je pro základní varianty mikropočítačů řady 48 nejvyšší dovolený, pro vyšší kmitočty by



Obr. 4. Náhrada krystalu 6 MHz (901–3)



Obr. 5. Jednoduchý převodník TTL/V.28 (901–5)

navíc bylo třeba modifikovat řídicí program. Při nižším kmitočtu by již mikroprocesor nestihl obsloužit příjem dat z počítače.

Výrobce dovoluje namísto krystalu zapojit i obvod podle **obr. 4.** Při přesném vyladění kmitočtu na 6 MHz by neměla být funkce interfejsu narušena. Problémy by však mohly nastat s časovou a teplotní stabilitou. Toto řešení nebylo experimentálně ověřeno.

Pro ty, kdo chtějí interfejs využít jako převodník na sériové rozhraní s napěťovou úrovní podle doporučení CCITT V.28 (RS 232C), je nutné ještě zapojení vybavit převodníkem TTL/V.28, např. podle **obr. 5.** Jiná řešení převodníku lze najít v [3]. Přitom je nutno zapojení ještě doplnit o záporné napájecí napětí (lze je získat např. z tiskárny). Optimální by samozřejmě bylo napájení +12 V, protože však doporučení V.28 definuje jako zakázanou oblast –3V až +3V, lze bez problémů použít napájení 5V.

Interfejs lze napájet přímo z počítače napětím + 5V (výrobce udává proudovou „rezervu" zdroje větší než 0,5 A), díky oddělenému napájecímu zdroji nehrozí poškození počítače ani při náhodném zkratu.

Protože originální třináctipólový konektor je téměř nesehnatelný (nejen u nás), je nutné tento konektor nahradit. Jako kontakty vyhoví např. dutinky z rozebraného konektoru CANON DB 25 (standard pro rozhraní V. 24) vlepené do destičky z organického skla s předvrtanými otvory v rastru 3,5 mm. Ještě lepší je však zabudovat do počítače např. pětikolíkový nf konektor (DIN), paralelně uvnitř počítače propojený na příslušné špičky třináctipólového konektoru. Tento konektor pak lze používat i pro připojení upraveného magnetofonu atd. Na straně tiskárny použijeme konektor podle typu tiskárny.

Řídicí program

Obsah EPROM s řídicím programem pro jednočipový mikropočítač je ve **Výpisu 1** (od adres nutno odečíst offset 8000H), obsah upraveného programu pro mikroprocesor 8039 (zapojení bez vnější RAM), je ve **Výpisu 2.**

Každý bajt z počítače je přijat rutinou pro zpracování interruptu – je na adresách 0AH až 40H. Po přijetí povelového bloku mikroprocesor zjistí, zda je oslovena tiskárna a pokud ano, vyšle na počítače příslušnou odpo-

věď přes vývod P26 a vyčká na datový blok. Ten je ukládán do RAM, poté se zjistí případný výskyt řídicích posloupností pro interfejs (viz dále) a nakonec je blok přenesen bajt po bajtu na port P1, který tvoří rozhraní CENTRONICS (porty 8048 jsou zakončeny budiči s otevřeným kolektorem a střádačem, což je pro tento účel výhodné). Vývod P24 mikroprocesoru se využívá pro vysílání potvrzovacích impulsů STB, vývod TO pro příjem signálu BUSY z rozhraní. Tyto dva signály spolu s osmi datovými budiči tvoří základní a postačující obvody rozhraní CENTRONICS. Je-li naprogramováno sériové vysílání, jsou bajty sériově vysílány přes vývod P25 zadanou přenosovou rychlostí. Po vyslání celého bloku interfejs oznámí počítači možnost pokračovat vysláním bajtu 43H na vývod P26. Je-li uvnitř přijatého bloku nalezen znak EOL, tj. 9BH, je nahrazen kombinací CR+LF (0DH+0AH). EOL je totiž označení konce logického řádku používané počítači ATARI. Protože počítač doplní blok obsahující EOL samými nulami až do 40 bajtů, jsou tyto nevýznamné nuly interfejsem ignorovány.

Máme-li k dispozici maticovou tiskárnu, která má sedm až devět jehliček a která využívá posloupnost *‹ESC›K* k přeřazení do grafického režimu (např. tiskárny EPSON a kompatibilní, novější maticové tiskárny ROBOTRON atd.), můžeme využít další funkce interfejsu. Při příjmu znaků s ASCII kódem 0 až 19H (0 až 31 dekadicky) umožňuje interfejs buď transparentní přenos těchto znaků (např. ASCII znak 14H=CTRL-N se používá pro tisk jednoho řádku zvětšenými písmeny), nebo přepne tiskárnu do grafického režimu a vytiskne přibližně stejný znak, jaký je zobrazen na stínítku (ATARI pseudografika). Každý znak je reprezentován šesti bajty – tzn. že je tisknou v matici 6×8, stejně jako alfanumerické znaky u tiskáren EPSON, na rozdíl od matice 8×8 na stínítku. Kódy pro grafické znaky jsou uloženy ve třetí stránce EPROM tak, že na adresách 300H až 305H je šest bajtů pro znak s kódem 1H atd. Tuto tabulku lze samozřejmě při programování EPROM předefinovat a vytvořit tak vlastní znaky, např. českou abecedu s háčky a čárkami. Spolu s možností vytvoření uživatelské znakové sady, kterou poskytuje počítač ATARI, máme tak účinný prostředek např. pro vytvoření vlastního textového editoru s českou abecedou a s možností tisku.

Přitom v grafickém režimu je možno předepsat, že se znak *ESCAPE* (kód 1BH) tiskne jako grafický znak nebo se interpretuje jako řídicí znak, což umožňuje používat řídicí posloupnosti pro tiskárnu (např. k přepínání druhů písma). V transparentním režimu je naopak možno předepsat, že interfejs všechny znaky s ASCII kódem 0 až 19H nahradí znakem ~ (kód 7EH), pokud je transparentní přenos nežádoucí.

Pro řízení interfejsu se využívá některých posloupností se znakem ESCAPE, které ne-

```
8000 44 BA 20 15 04 0A 31 37 38 20 C5 AE 00 00 00 27
8010 AB 86 15 04 3B BC 08 B9 03 46 21 BF 05 B8 20 04
8020 27 D5 BD 29 B8 00 C5 E9 27 97 86 2F A7 04 32 00
8030 00 00 FB 67 AB B9 03 00 EC 27 A5 FE 86 3C 05 93
8040 80 D3 72 96 51 18 FF 53 F8 AF 80 53 F8 5F AF 04
8050 9E 80 D3 6A 96 5C FF 53 F7 AF 04 9E 80 D3 69 96
8060 67 FF 43 08 AF 04 9E 80 D3 63 96 72 FF 43 40 AF
8070 04 9E 80 D3 7A 96 7D FF 53 EF AF 04 9E 80 D3 64
8080 96 88 FF 43 20 AF 04 9E 80 D3 6E 96 93 FF 53 DF
8090 AF 04 9E 80 D3 67 C8 96 A1 18 FF 53 BF AF 97 A7
80A0 83 97 83 2E 42 59 54 55 0D 0A 20 20 30 31 33 38
80B0 20 42 45 34 31 20 20 20 20 20 20 20 20 20 31 38
80C0 32 20 20 20 20 20 20 20 20 20 4D 4F 56 20 52 36
80D0 2C 23 34 31 48 0D 0C 0A 0A 0A 49 53 49 53 2D 49
80E0 49 20 4D 43 53 2D 34 38 2F 55 50 49 2D 34 31 20
80F0 4D 41 43 52 4F 20 41 53 53 45 4D 42 4C 45 52 2C
8100 76 00 B5 C5 56 30 FB A0 18 EF 00 34 9E 96 00 46
8110 0F 85 BF 05 23 20 A8 D5 17 A9 F1 D3 53 C6 25 BE
8120 41 54 95 24 00 B8 F8 B9 07 54 8C B8 00 95 24 00
8130 FB D5 90 18 ED 00 BD 29 BE 41 B6 00 95 BB FF EB
8140 3F 54 95 B8 00 B9 28 8A 10 9A F8 80 D3 9B C6 94
8150 FF 37 92 62 80 D3 1B 96 62 18 14 40 E6 62 E9 8F
8160 24 96 FF D2 7F B2 6C 80 D3 1B C6 8C 54 4B F6 8F
8170 80 53 7F 03 E0 F6 8C 54 61 54 71 54 7F 24 8F FF
8180 B2 8C 80 53 7F 03 E0 F6 8C 23 7E 90 80 54 00 18
8190 E9 4B 24 96 54 42 BE 43 54 95 B8 00 24 00 BF 05
81A0 B8 20 F0 D3 40 83 20 53 55 50 45 52 20 20 20 20
81B0 20 20 20 3B 46 30 3D 4E 45 4E 49 20 54 4F 20 50
81C0 52 4F 20 4C 50 3A 0D 0A 20 20 30 31 33 43 20 39
81D0 35 20 20 20 20 20 20 20 20 20 20 20 31 38 34 20
81E0 20 20 20 20 20 20 20 20 43 50 4C 20 46 30 20 20
81F0 20 20 20 20 20 20 20 20 3B 4A 45 20 54 4F 20 50
8200 AE FF 72 08 80 53 7F 90 FF 53 07 96 16 36 0D FE
8210 39 9A EF 8A 10 83 FE 35 9A DF 54 37 BA 08 00 00
8220 67 F6 27 9A DF 44 2B 8A 20 00 00 54 37 EA 20 00
8230 00 00 8A 20 54 37 83 AE FC 62 55 FE 16 40 44 3C
8240 65 83 23 0D 54 00 23 0A 54 00 83 97 80 D3 A0 96
8250 60 FF 37 72 60 A7 54 61 23 FF BB 06 54 00 EB 5C
8260 83 23 1B 54 00 23 4B 54 00 23 06 54 00 27 54 00
8270 83 80 97 85 F7 F6 78 95 AB E7 6B BB 06 AD 83 FD
8280 E3 B6 84 37 54 00 1D EB 7F BD 29 83 F8 A3 AE 54
8290 95 18 E9 8C 83 9A BF BB 04 00 EB 9A BA 08 FE BB
82A0 05 67 F6 A8 9A BF 44 AC 8A 40 00 00 EB AC EA 9F
82B0 00 00 00 BB 08 8A 40 EB B7 83 15 23 70 3A 27 C5
82C0 BF 05 AE AD AC AB AA A9 B8 20 D5 AF AE BD 29 AC
82D0 AB AA A9 A8 8A 80 09 AF 9A 7F 85 95 A5 B5 35 FF
82E0 53 07 03 F0 A3 AC C5 05 24 00 2E 41 44 52 45 53
82F0 00 07 84 C2 D7 EC F6 00 41 43 00 00 14 29 3D 20
8300 00 38 7C 3E 7C 38 00 00 FF FF 18 18 00 00 00 00
8310 FF FF 18 18 F8 F8 00 00 18 18 FF FF 00 00 18 18
8320 1F 1F 00 00 07 0E 1C 38 70 E0 E0 70 38 1C 0E 07
8330 03 07 0F 1F 3F FF 00 00 00 0F 0F 0F FF 3F 1F 0F
8340 07 03 00 00 00 F0 F0 F0 F0 F0 F0 00 00 00 C0 C0
8350 C0 C0 C0 C0 03 03 03 03 F0 F0 F0 00 00 00 00 00
8360 00 1D 77 63 77 1D 00 00 1F 1F 18 18 18 18 18 18
8370 18 18 18 18 FF FF 18 18 00 18 3C 3C 18 00 0F 0F
8380 0F 0F 0F 0F FF FF 00 00 00 00 18 18 1F 1F 18 18
8390 18 18 F8 F8 18 18 FF FF FF 00 00 00 00 00 F8 F8
83A0 18 18 00 7C 54 47 05 00 00 18 30 7E 30 18 00 18
83B0 0C 7E 0C 18 08 08 2A 3E 1C 08 08 1C 3E 2A 08 08
83C0 56 20 52 41 4D 0D 0A 20 20 30 31 34 42 20 38 30
83D0 20 20 20 20 20 20 20 20 20 20 20 31 39 32 20 54
83E0 49 53 4B 31 3A 20 20 4D 4F 56 58 20 41 2C 40 52
83F0 30 0D 0A 20 20 30 31 34 43 20 44 33 39 42 20 20
```

Výpis 1. *Obsah ROM pro mikroprocesory 8035, 8048, 8748* (901–V1)

```
8000 44 BA 8D 15 04 0A 36 92 1F D2 C5 AE 00 00 00 27
8010 AB 86 15 04 3B BC 08 B9 03 46 21 BF 05 B8 20 04
8020 27 D5 BD 29 B8 40 C5 E9 27 97 86 2F A7 04 32 00
8030 00 00 FB 67 AB B9 03 00 EC 27 A5 FE 86 3C 05 93
8040 F0 D3 72 96 51 18 FF 53 F8 AF F0 53 F8 5F AF 04
8050 9E F0 D3 6A 96 5C FF 53 F7 AF 04 9E F0 D3 69 96
8060 67 FF 43 08 AF 04 9E F0 D3 63 96 72 FF 43 40 AF
8070 04 9E F0 D3 7A 96 7D FF 53 EF AF 04 9E F0 D3 64
8080 96 88 FF 43 20 AF 04 9E F0 D3 6E 96 93 FF 53 DF
8090 AF 04 9E F0 D3 67 C8 96 A1 18 FF 53 BF AF 97 A7
80A0 83 97 83 7E 22 BA A8 CD 5F 7F C9 21 AE 8D 22 B7
80B0 A8 2B 7E 32 B9 A8 11 C3 8D 01 68 7E CD BD 84 11
80C0 C3 8D 01 AD 8D CD E8 83 FE 00 C2 09 81 21 68 A8
80D0 36 03 11 39 92 01 C4 8D CD 32 8C 3E 00 11 39 92
80E0 CD 08 8C B5 D6 01 9F F5 3E 0D 1B CD 08 8C B5 D6
80F0 01 9F C1 48 B1 1F D2 06 81 01 AE 8D C5 11 39 92
8100 76 00 B5 C5 56 30 FB A0 18 EF 00 34 9E 96 00 46
8110 0F 85 BF 05 23 20 A8 D5 17 A9 F1 D3 53 C6 25 BE
8120 41 54 95 24 00 B8 F8 B9 07 54 8C B8 40 95 24 00
8130 FB D5 A0 18 ED 00 BD 29 BE 41 B6 00 95 BB FF EB
8140 3F 54 95 B8 40 B9 28 8A 10 9A F8 F0 D3 9B C6 94
8150 FF 37 92 62 F0 D3 1B 96 62 18 14 40 E6 62 E9 8F
8160 24 96 FF D2 7F B2 6C F0 D3 1B C6 8C 54 4B F6 8F
8170 F0 53 7F 03 E0 F6 8C 54 61 54 71 54 7F 24 8F FF
8180 B2 8C F0 53 7F 03 E0 F6 8C 23 7E A0 F0 54 00 18
8190 E9 4B 24 96 54 42 BE 43 54 95 B8 40 24 00 BF 05
81A0 B8 20 F0 D3 40 83 23 72 0E 06 CD 14 7E 3E 00 11
81B0 39 92 CD 08 8C B5 CA BA 81 C9 2A C4 A8 E5 2A A7
81C0 8D 44 4D 2A 9D 8D EB CD 2A 7E 3E 00 11 39 92 CD
81D0 08 8C B5 CA D7 81 C9 2A C5 A8 2B 22 C5 A8 C3 88
81E0 81 CD 81 72 C9 21 C7 A8 71 2A C7 A8 4D 1E 02 CD
81F0 3C 7D 3E 00 11 39 92 CD 08 8C B5 CA FF 81 C9 2A
8200 AE FF 72 08 F0 53 7F A0 FF 53 07 96 16 36 0D FE
8210 39 9A EF 8A 10 83 FE 35 9A DF 54 37 BA 08 00 00
8220 67 F6 27 9A DF 44 2B 8A 20 00 00 54 37 EA 20 00
8230 00 00 8A 20 54 37 83 AE FC 62 55 FE 16 40 44 3C
8240 65 83 23 0D 54 00 23 0A 54 00 83 97 F0 D3 A0 96
8250 60 FF 37 72 60 A7 54 61 23 FF BB 06 54 00 EB 5C
8260 83 23 1B 54 00 23 4B 54 00 23 06 54 00 27 54 00
8270 83 F0 97 85 F7 F6 78 95 AB E7 6B BB 06 AD 83 FD
8280 E3 B6 84 37 54 00 1D EB 7F BD 29 83 F8 A3 AE 54
8290 95 18 E9 8C 83 9A BF BB 04 00 EB 9A BA 08 FE BB
82A0 05 67 F6 A8 9A BF 44 AC 8A 40 00 00 EB AC EA 9F
82B0 00 00 00 BB 08 8A 40 EB B7 83 15 23 70 3A 27 C5
82C0 BF 05 AE AD AC AB AA A9 B8 20 D5 AF AE BD 29 AC
82D0 AB AA A9 B8 40 8A 80 09 AF 9A 7F 85 95 A5 B5 35
82E0 FF 53 07 03 F0 A3 AC C5 05 24 00 8D CD 0A 46 01
82F0 00 07 84 C2 D7 EC F6 00 41 43 00 00 14 29 3D 95
8300 00 38 7C 3E 7C 38 00 00 FF FF 18 18 00 00 00 00
8310 FF FF 18 18 F8 F8 00 00 18 18 FF FF 00 00 18 18
8320 1F 1F 00 00 07 0E 1C 38 70 E0 E0 70 38 1C 0E 07
8330 03 07 0F 1F 3F FF 00 00 00 0F 0F 0F FF 3F 1F 0F
8340 07 03 00 00 00 F0 F0 F0 F0 F0 F0 00 00 00 C0 C0
8350 C0 C0 C0 C0 03 03 03 03 F0 F0 F0 00 00 00 00 00
8360 00 1D 77 63 77 1D 00 00 1F 1F 18 18 18 18 18 18
8370 18 18 18 18 FF FF 18 18 00 18 3C 3C 18 00 0F 0F
8380 0F 0F 0F 0F FF FF 00 00 00 00 18 18 1F 1F 18 18
8390 18 18 F8 F8 18 18 FF FF FF 00 00 00 00 00 F8 F8
83A0 18 18 00 7C 54 47 05 00 00 18 30 7E 30 18 00 18
83B0 0C 7E 0C 18 08 08 2A 3E 1C 08 08 1C 3E 2A 08 08
83C0 39 92 CD 08 8C B5 C2 CC 83 CD A7 82 CD 2B 60 C3
83D0 DB 83 CD 95 84 CD A7 82 CD 2B 60 C9 0A 3C C5 42
83E0 4B 5F 16 00 CD FE 8A C9 EB 0A 5F 1C 0B 2B 03 23
83F0 0A BE C2 FC 83 1D C2 EE 83 3E FF C9 AF C9 2A B5
```

Výpis 2. *Obsah ROM pro mikroprocesor 8039* (901–V2)

jsou běžnými tiskárnami využívány. Jejich přehled spolu s významem je v **tabulce 1.**

Po zapnutí je interfejs inicializován tak, že je transparentní pro všechny znaky a výstup je na port T1, tj. rozhraní CENTRONICS. Chceme-li, aby výstup z interfejsu byl sériový rychlostí např. 50 Bd, aby znaky s kódem menším než 20H byly tisknuty v grafickém režimu včetně znaku ESCAPE, a inverzní alfanumerické znaky mají být tisknuty jako normální (nejvyšší bit nastaven do 0), zadáme jako první příkaz po zapnutí počítače:

LPRINT „‹ESC›r"; CHRS (1); „‹ESC›j‹ESC›d‹ESC›g"

Chceme-li, aby interfejs pracoval v režimu CENTRONICS, přičemž znaky s kódem nižším než 20H mají být nahrazeny vlnovkou a interfejs má v tomto režimu trvale zůstat, zadáme po zapnutí:

LPRINT „‹ESC›r"; CHRS (0); „‹ESC›n‹ESC›c‹ESC›z"

Konečně chceme-li, aby interfejs pracoval v režimu CENTRONICS, přičemž znaky s kódem nižším než 20H mají být interpretovány jako grafické a zároveň chceme mít možnost používat řídicí posloupnosti pro tiskárnu, tzn. že ESCAPE nesmí být interpretován jako grafický znak, zadáme:

LPRINT „‹ESC›r"; CHRS (0); „‹ESC›n‹ESC›g"

Podobných příkladů lze samozřejmě vymyslet mnohem více.

Pozn.: znak ‹ESC› (ESCAPE) dostaneme, stiskneme-li na počítači 2× za sebou klávesu ESC.

Závěrem bych chtěl poznamenat, že pokud je mi známo, nebyl dosud nikde zveřejněn podrobný popis komunikace počítačů ATARI s periferiemi. Zde zveřejněný popis je výsledkem experimentální práce, analýzy přenosu pomocí datového analyzátoru a simulátoru a disassemblování příslušných rutin operačního systému. Proto nejsou pravděpodobně ošetřeny úplně všechny možné situace, které se při přenosu mohou teoreticky vyskytnout a které nebylo možno nasimulovat. Ze stejného důvodu jsou interfejsem ignorovány zabezpečovací součtové bajty. Pro toto řešení však mluví praxe: Za zhruba půl roku provozu interfejsu nebyla zaregistrována ani jedna chyba při přenosu i při vysoké přenosové rychlosti, která je počítači ATARI používána.

Tab. 1. Přehled řídicích posloupností pro interfejs (901–T1)

Řídicí posloupnost	Hexa-decimálně	Nastavení interfejsu
<ESC>r CTRL -.	1BH,72H,00H	paralelní výstup z interfejsu (CENTRONICS)
<ESC>r CTRL -A	1BH,72H,01H	sériový výstup z interfejsu 50 Bd
<ESC>r CTRL -B	1BH,72H,02H	" 100 Bd
<ESC>r CTRL -C	1BH,72H,03H	" 200 Bd
<ESC>r CTRL -D	1BH,72H,04H	" 300 Bd
<ESC>r CTRL -E	1BH,72H,05H	" 600 Bd
<ESC>r CTRL -F	1BH,72H,06H	" 1200 Bd
<ESC>j	1BH,6AH	nastavení nejvyššího bitu přenášených znaků do log.0
<ESC>i	1BH,69H	transparentní přenos nejvyššího bitu
<ESC>d<ESC>c	1BH,64H 1BH,63H	transparentní přenos všech znaků
<ESC>d<ESC>g	1BH,64H 1BH,67H	všechny znaky s kódem nižším než 20H interpretovány jako grafické včetně<ESC>
<ESC>n<ESC>c	1BH,6EH 1BH,63H	všechny znaky s kódem nižším než 20H nahrazovány vlnovkou (ASCII kód 7EH)
<ESC>n<ESC>g	1BH,6EH 1BH,67H	všechny znaky s kódem nižším než 20H interpretovány jako grafické, <ESC> přenášen transparentně
<ESC>z	1BH,7AH	zákaz interpretace řídicích posloupnosti z této tabulky, používá se při výpisu paměti apod., aby nemohlo dojít k nežádoucímu přepnutí režimu interfejsu náhodným příjmem posloupností znaků stejné jako některá řídicí posloupnost

INTERFEJS PRO MAGNETOFON K ATARI

Ing. Milan Kuchař, *739 44 Brušperk 932*

V AR A8/87 byl uveřejněn popis interfejsu k magnetofonu pro ATARI 800XL, který byl téměř přesnou kopií zapojení speciálního firemního magnetofonu XC12. Vzhledem k napájecímu napětí 5 V zde není možné bez problémů použít jakékoliv běžné dostupné operační zesilovače. Např. typy 741, 748, 1458 mají nejnižší úroveň na výstupu v nesymetrickém zapojení asi 2 V, což je pro dané zapojení nevyhovující. Z tuzemských OZ je proto vhodné použít např. typy MAB355,6,7 nebo MAC155,6,7. Dále je zapojení poměrně citlivé na výběr součástek v pásmových propustích. Proto jsem vyvinul zapojení interfejsu pracující na jiném principu – s fázovým závěsem.

Literatura

[1] *Eichler, Grohmann:* ATARI INTERN, DATA BECKER 1984.

[2] *Intel Component Data Catalog, Intel Corporation 1981.*

[3] *Hyan, J., T.:* RS 232C – V. 24 AR A10/84.

Seznam součástek

Mikroprocesor	MHB8048, MHB8035, MHB8748, I8039 nebo ekvivalenty
RAM	2× MHB2114, HM6116 nebo jiná statická RAM
EPROM	MHB8708C, I2716, nebo ekvivalenty
Latch	MHB8282, SN74LS373 nebo jiný osmibitový paralelní střádač
Hradla	MH7400

Popis zapojení

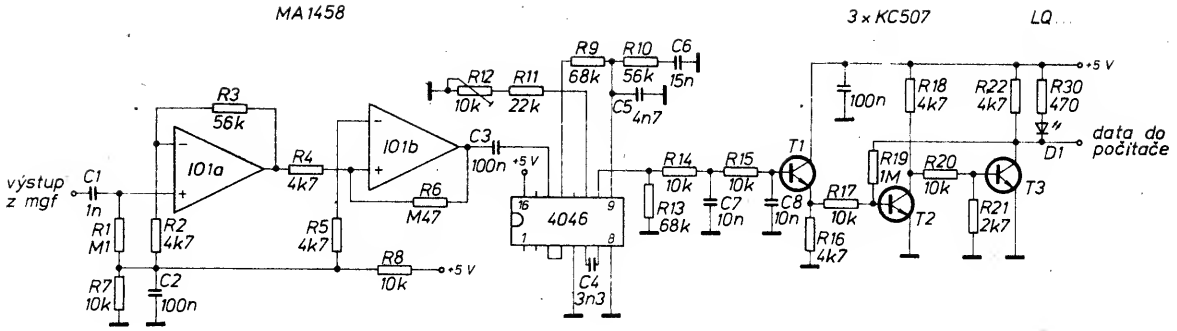
Záznam informace na kazety u systému ATARI je prováděn tak, že logická „1" je zaznamenávána kmitočtem 5135 Hz, logická „0" kmitočtem 3995 Hz a data jsou vysílána rychlostí 600 Bd v sérii s jedním nulovým start-bitem a jedním jedničkovým stop-bitem bez parity. Zaváděcí kmitočet je 5135 Hz – tedy logická „1". Zapojení interfejsu je na **obr. 1.** IO1a předzesiluje signál z magnetofonu, komparátor IO1b převádí signál na obdélníky. Z jeho výstupu je signál veden přes C3 na vstup fázového komparátoru PLL (14) (obvod 4046 kondenzátorovou vazbu povoluje). Výstup fázového komparátoru II (13) je veden přes filtr (R9,R10,C5,C6) na vstup VCO (9) a dále z vnitřního emitorového sledovače obvodu 4046 (10) přes dolní propust (T14,R15,C7,C8) na T1, který funguje jako transformátor impedance. Z emitoru T1 je signál veden na napěťový komparátor s hysterezí – T2,T3, na jehož výstupu je hotový signál pro počítač. Popis funkce fázového závěsu neuvádím, zájemci jej mohou najít např. v [2].

Signál z počítače do magnetofonu je nutno zeslabit a omezit vyšší harmonické kmitočty – **viz obr. 2.** Trimrem R24 nastavujeme vstupní napětí pro použitý druh magnetofonu.

Postup při oživování

Při použití ověřených součástek interfejs pracuje na první zapojení. Svítivá dioda D1 rozsvícením signalizuje logickou „0" a zhasnutím logickou „1".

Seřízení provedeme tak, že nejdříve do interfejsu pustíme z magnetofonu zaváděcí signál některého programu (je před každým

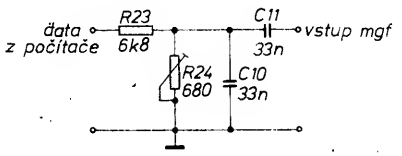


Obr. 1. Schéma zapojení interfejsu (907–1)

programem a trvá 20 s) a trimrem R12 otáčíme tak, aby se svítivá dioda D1 právě rozsvítila; tuto polohu trimru si označíme. Pak pustíme do interfejsu z magnetofonu program a trimrem R12 otáčíme na druhou stranu, až D1 právě přestane blikat. Poté nastavíme R12 doprostřed mezi tyto dvě polohy a seřizování je ukončeno. Jestliže se nám nepodaří najít tyto dvě polohy trimru R12, je nutno změnit odpor rezistoru R11 nebo kapacitu kondenzátoru C4.

Poznámky ke konstrukci

Návrh plošných spojů neuvádím, protože zařízení je tak jednoduché a malé, že je možno je umístit přímo do magnetofonu. Trimr R12 je vhodné umístit přístupně – je jím možno korigovat rozdílnost otáček při nahrávání programů pořízených na jiném magnetofonu.



Obr. 2. Úprava signálu pro magnetofon (907–2)

Seznam součástek

Rezistory TR 151:

R1	100 kΩ
R2,R4,R5,R16,	
R18,R22	4,7 kΩ
R3,R10	56 kΩ
R6	470 kΩ
R7,R8,R14,R15,	
R17,R20	10 kΩ
R9,R13	68 kΩ
R11	22 kΩ
R19	1 MΩ
R21	2,7 kΩ
R23	6,8 kΩ
R30	470 Ω

Potenciometrické trimry:

R12	10 kΩ
R24	680 Ω

Kondenzátory keramické:

C1	1 nF
C2,C3,C9	100 nF
C4	3,3 nF
C5	4,7 nF
C6	15 nF
C7,C8	10 nF
C10,C11	33 nF

Polovodičové prvky:

IO1	MA1458
IO2	MHB4046
T1,T2,T3	KC507 až KC509
D1	libovolná svítivá doba

INTERFEJS ATARI 800 – ALFIGRAF

Ing. Pavel Rada, *Mezihorská 56, 140 00 Praha 4*

Krokové reverzační motory typu SMR 300-100/24, případně SMR 300-300/24 umožňují stavbu relativně jednoduchých a přitom spolehlivých periferních zařízení k počítačům. Jednou z atraktivních aplikací jsou zapisovače, umožňující jak tisk, tak kresbu grafiky.

Příkladem může být jak tovární konstrukce zapisovače MINIGRAF 0507, tak zapisovač ALFI sestavený ze stavebnice Merkur podle návrhu ing. Dovala.

Zapojení interfejsů k těmto typům zapisovačů bylo zatím publikováno především pro připojení na počítače kompatibilní se ZX Spectrum, kde se využívá obvodu MHB 8255.

Osmibitové Atari mají již obvod vykonávající základní funkce interfejsu zabudovaný uvnitř (PIA) a vyvedený na dva ovladačové konektory. To umožňuje stavět značně jednodušší konstrukce. Například pákové ovladače pro Atari obsahují jen spínací kontakty.

Programově lze přes ovladačové kontakty vyvést i osmibitovou sběrnici a umožnit tak snadné a jednotné připojení periférií k celé typové řadě osmibitových Atari.

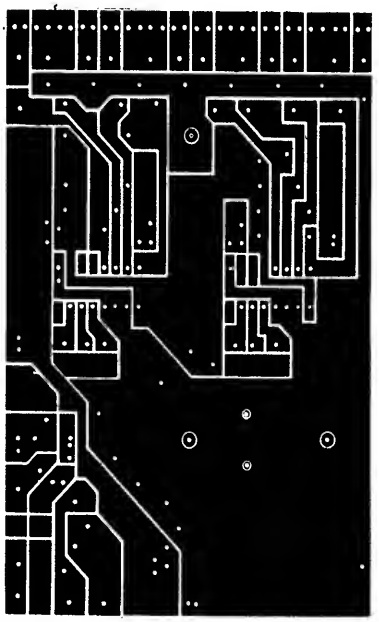
Konstrukční řešení navazující na uspořádání zapisovače ALFI a využívající programového vybavení ing. Jandíka pro MINIGRAF bylo nazváno ALFIGRAF. Jeho zapojení je na **obr. 1.**

Soubor programů v systému Turbo 2000 pro zapisovače typu ALFIGRAF nebo Minigraf, včetně textového editoru, byl sestaven na magnetofonovou kazetu typu C60, kde zaplňuje téměř celou jednu stranu a obsahuje souhrně přes 2×150 kB dat.

Zápis na dodanou kazetu zajišťuje i nečlenům klubu, proti proplacení složenky (100 Kčs), S.W. skupina klubu, a lze jej objednávat zasláním korespondenčního lístku s označením SYKO 007/89 na adresu:

487. ZO SVAZARMU ATARI KLUB PRAHA, pošt. přihrádka 51, 100 00 Praha 10.

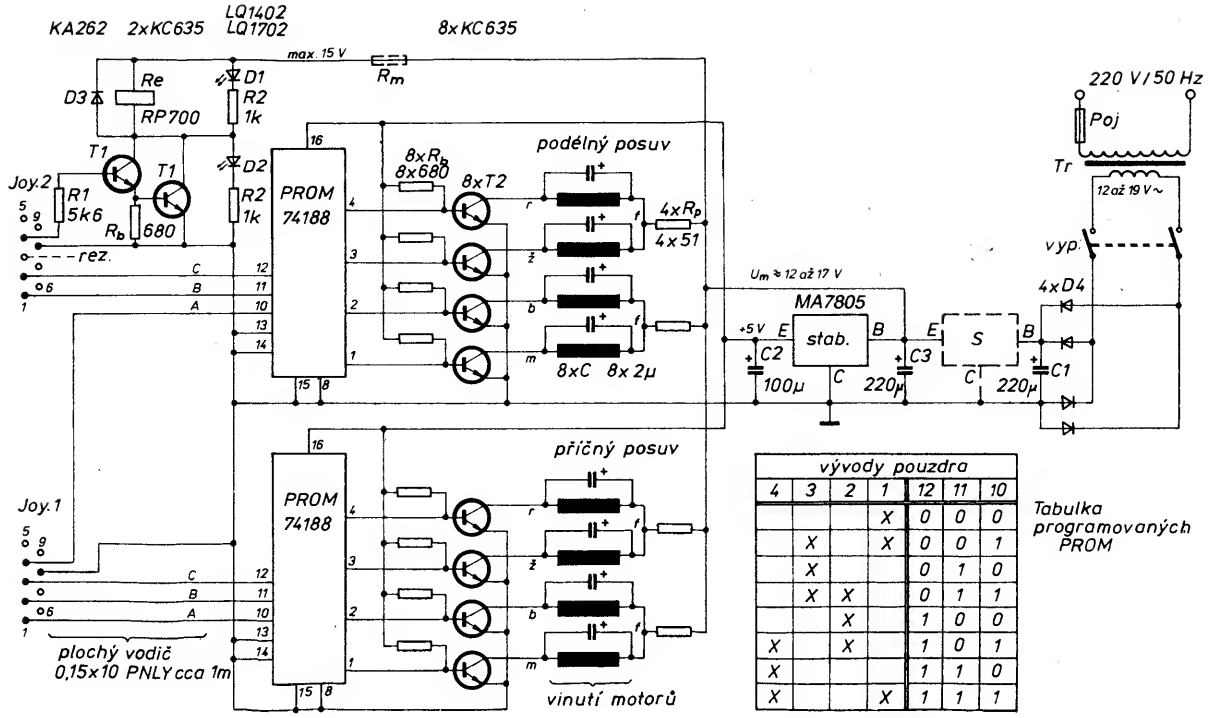
Pro uvedené zapojení je nezbytné použít naprogramovanou bipolární paměť PROM MH74188. Způsob naprogramování, při kterém je využita jen malá část kapacity paměti, je uveden na **obr. 2**, kde křížky označují logickou hodnotu 1 při zapojení otevřených kolektorů přes rezistor 680 Ω na plus 5 V.



Obr. 2. Obrazec plošných spojů desky Y505 (906–2)

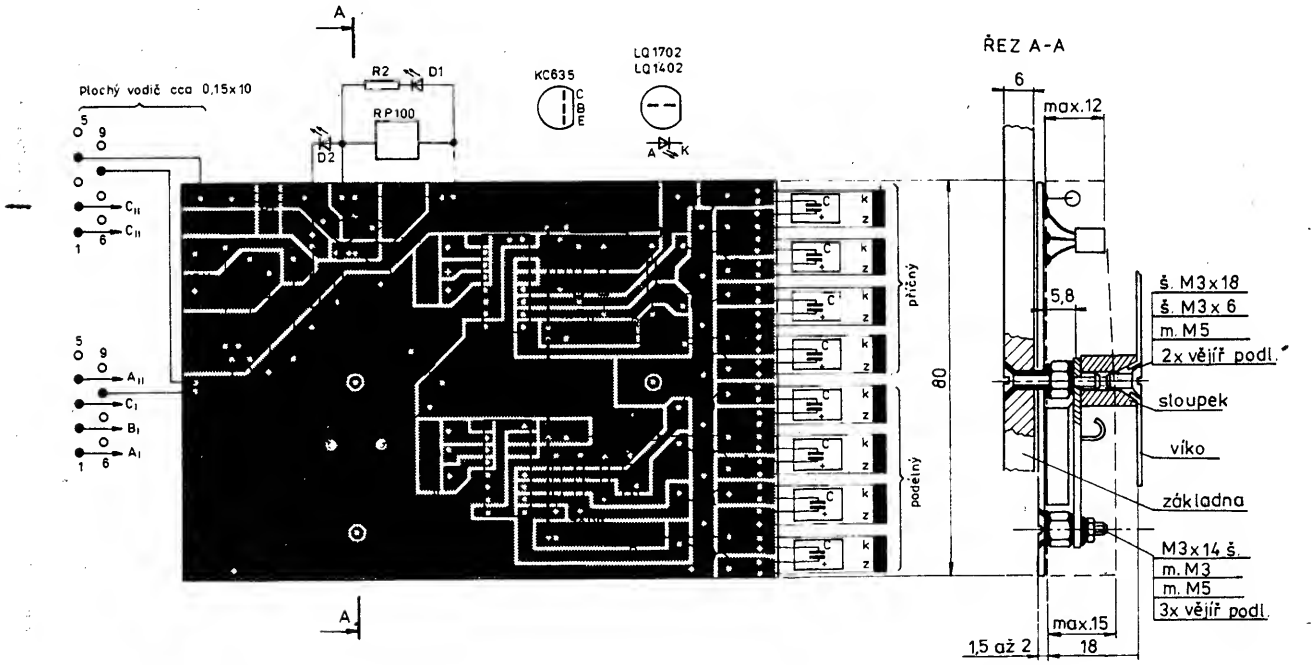
Zapojení tohoto typu je odolné proti změně napájecích napětí a variabilitě součástek. Po montáži proto postačí pečlivá vizuální kontrola správnosti zapojení konektorů a obvodu na desce s plošnými spoji a při oživování postačí měřicí přístroj typu Avomet.

Deska na **obr. 3** je jednostranná a plošné spoje jsou vytvořeny dělicími čarami. Součástky jsou pájeny ze strany spojů. Odpadá tak vrtání otvorů, zvýší se přehlednost a variabilnost pro použití různých náhrad za zá-



4	3	2	1	12	11	10
			X	0	0	0
	X		X	0	0	1
	X			0	1	0
	X	X		0	1	1
		X		1	0	0
X		X		1	0	1
X				1	1	0
X			X	1	1	1

Obr. 1. Schéma Alfigrafu pro Atari 800XL/XE, 130XE (906–1)



Obr. 3. Rozmístění součástek na desce Y505 a celkové konstrukční uspořádání. Všechny součástky se pájejí ze strany mědi. Vývody PROM č. 5, 6, 7 a 9 nezapojujte! Odehněte je do vzduchu! (906–3)

kladní typy součástek. Velké plochy mědi pomáhají odvádět teplo, kterého se uvolňuje nejvíce z rezistorů i při klidovém stavu motorů.

Rozměrově je deska určena k zabudování do ALFIGRAFU, ale při zajištění odvodu tepla ze stabilizátoru jiným způsobem než přes sloupek na víko kryjící elektroniku lze zapojení použít i pro jiná konstrukční řešení.

Použité podklady

Návod ze stavebnic Merkur – ALFI.
Návod k obsluze MINIGRAFu a jeho modulu pro Atari.
Zpravodaje 487. ZO Svazarmu č. 1; 4; 6/1989.

Použité součástky

S	pro $U_{sek} > 17$ (14) V použít stabilizátor MA7812(7815)
T1	KC635 $h_{21E} > 160$ nebo Darlington
T2	KC635 (KC637, K4639)
C	2 µF/70 V TE006
C1, C3	220 µF/40 V TF010
C2	100 µF/6 V TE 981
R1	4,7 kΩ i více podle zesílení T1
R2	1 kΩ (v sérii s D1 může být jen 510 Ω
R_b	680 Ω i více
R_m	předřadný odpor, použít pro $U_m > 15$ V
R_p	33 až 51 Ω/2 – 4 W volit dle U_m
D1	LQ1402 žlutá
D2	LQ1702 zelená
D3	KA262 nebo KY13
D4	KY132/80
Tr	transformátor min. 20 W
Motory	SMR 300-100 RI/24 (r – rudá, f – fialová, b – bílá, m – modrá, ž - žlutá)

AUTOMATIZOVANÝ EXPOZIČNÝ SYSTÉM

Ing. Ivan Hejda, *Jašíkova 15, 821 03 Bratislava* **Ing. Ján Kačmárik,** *nám. Ľud. milície 10, 821 09 Bratislava*

Nasledujúce zapojenie spolu s uvedeným programovým vybavením značne zľahčuje práce vo fotokomore pri čiernobielom pozitívnom procese. Je pripojené na mikropočítač ZX-Spectrum cez paralelný interfejs s MHB8255 (AR 6/85), plne nahradzuje osvitomer a spínacie hodiny. Na základe nameraného jasu snímky vypočítava expozičnú dobu, meria teplotu vyvolávacieho kúpeľa, resp. prepieracej vody. Všetky operácie vrátane spínania zväčšovacieho prístroja vykonáva užívateľ prostredníctvom klávesnice mikropočítača a mikrospínača. Spoluprácu s obslužným programom zľahčujú pomocné údaje na obrazovke, ktorá je kvôli zatemneniu prikrytá červenou fóliou. Užívateľ má možnosť použiť vypočítanú dobu expozície, prípadne ju opraviť podľa vlastného uváženia a opakovať expozíciu pri opakovanom zväčšovaní tej istej snímky.

Podstatnú časť zapojenia tvoria snímač intenzity osvetlenia a výkonový spínač.

Použitie zariadenia je programovým vybavením viazané na mikropočítač ZX-Spectrum. Po prepísaní programu je použiteľné s ľubovoľným mikropočítačom, ktorý je vybavený paralelným interfejsom.

Hlavné použité súčiastky: E555, WK65061, 11NR15, MHB4011, impulzný transformátor.

Popis konštrukcie

Schéma zapojenia je na **obr. 1,** obrazec plošných spojov na **obr. 2** a zapojenie konektorov k paralelnému interfejsu a snímačom na **obr. 4.**

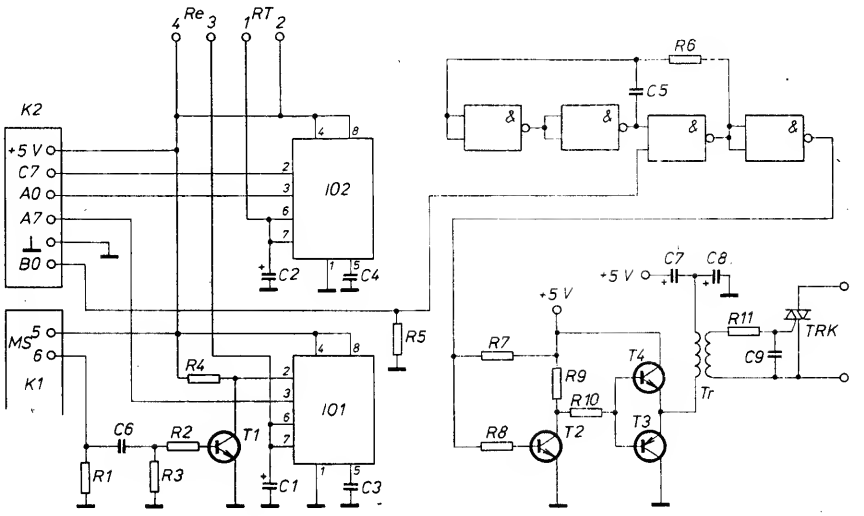
Výkonový spínač pre ovládanie zväčšovacieho prístroja je riadený bitom B0 MHB8255. Časť merania teploty vývojky resp. prepieracej kúpele s IO E555 je identická s prevodníkom (AR 6/88). Štartovací impulz prevodu je generovaný mikropočítačom cez bit C7 MHB8255, výstupný impulz prevodníka je pripojený na bit A0. Termistor 11NR15 je cez ohybný vodič pripojený k bodom 1,2 zapojenia, aby sa dal umiestniť priamo do média, teplotu ktorého meriame.

Snímač jasu snímky s IO1 pracuje na tom istom princípe s tým, že štartovací impulz prevodu nedáva mikropočítač, ale obsluha stlačením mikrospínača pripojeného k bodom 5,6 zapojenia. Samotný fotorezistor WK 65061 je cez ohybný vodič pripojený k bodom 3,4. Je umiestnený spolu s mikrospínačom na jednoduchom držadle (celok v ďalšom texte nazývame sondou). Obsluha takto môže umiestniť fotoodpor do želaného miesta exponovanej snímky (ako pri osvitomere) a mikrospínačom odštartovať prevod. U oboch snímačov je prevod 16-bitový s ochranou proti preplneniu počítadla.

Popis obsluhy

Ovládanie AES zabezpečuje program v jazyku BASIC **(Výpis 1)** využívajúci spomínané snímače pomocou podprogramu v strojovom kóde **(Výpis 2).** Výkonový spínač je riadený inštrukciou OUT.

Po spustení program núka užívateľovi 5 možností, ktoré sa volia stlačením týchto klávesov:



Obr. 1. Schéma zapojenia (911–1)

"t"– expozičný test: tento je treba vykonať, ak chce užívateľ používať automatické určovanie doby expozície (zmienime sa o ňom neskôr),

"k"– zmena expozičnej konštanty: túto užívateľ vykoná ak sa zmenia podmienky vyvolávania resp. podľa svojho uváženia (o tejto možnosti sa tiež zmienime neskôr).

"e"– skok do expozičného bloku, ktorý je jadrom práce s AES.

"f"– meranie teploty vývojky resp. prepieracieho kúpeľa. Automaticky se zmeria a zobrazí príslušná teplota s presnosťou na 0,1 °C.

"c"– podprogram na ciachovanie termistora. Toto ciachovanie vykonáme hneď pri prvom použití AES, pričom namerané konštanty sa uložia na pásku za samotný obslužný program. Pri ďalšom používaní už netreba teplomer ciachovať, príslušné konštanty sa nahrajú z pásky.

Po voľbe expozičného bloku sa zobrazí "menu" ponúkajúce tieto možnosti:

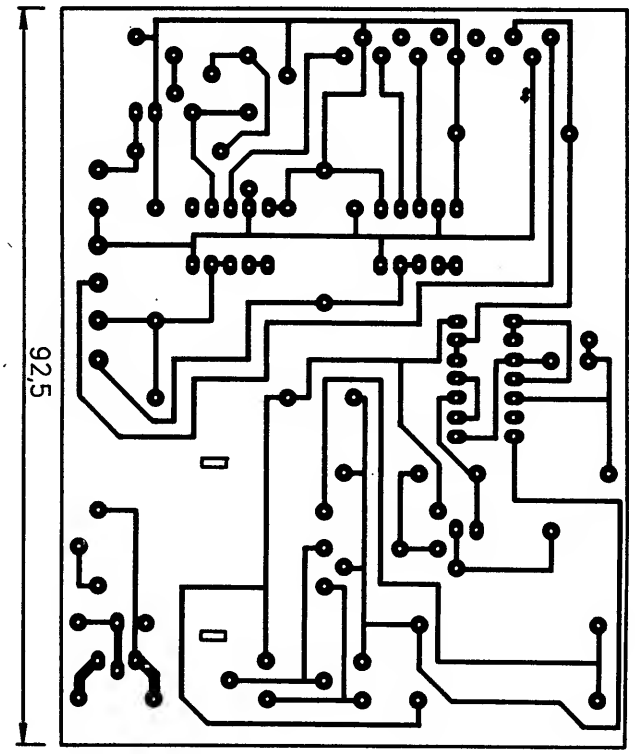
"Z"– zadanie požadovanej doby expozície v sekundách z klávesnice.

"B"– meranie bodovým snímačom. Pri zvolení tejto činnosti sa predpokladá, že v ráme zväčšovacieho prístroja je pripravený negatív snímky. Automaticky sa zapne zväčšovací prístroj. Užívateľ si vyberie ľubovoľný bod snímky a zadá cez klávesnicu aký stupeň sčernania má mať pozitív v tomto mieste. Stupeň sčernania sa volí podľa stupnice šedi, o ktorej sa zmienime neskôr. Potom položí sondu na zvolené miesto a mikrospínačom odštartuje meranie. Po skončení prevodu sa automaticky vypne zväčšovací prístroj, vypočíta a zobrazí sa doba expozície.

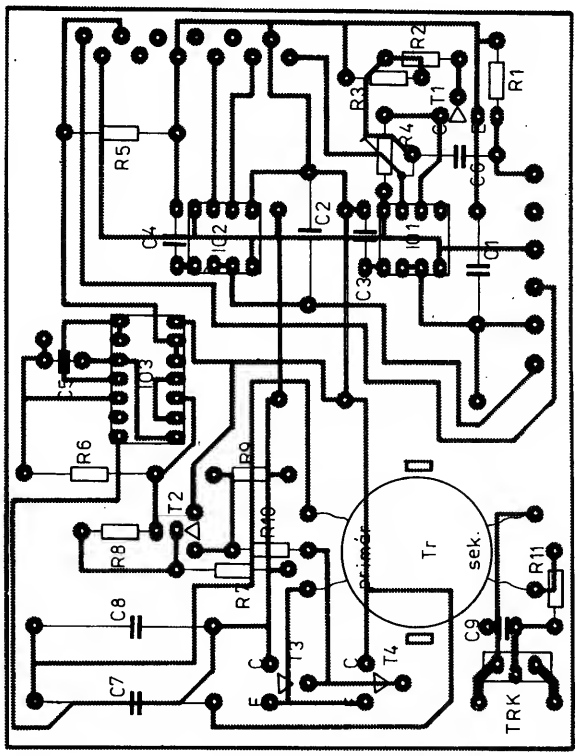
"E"– expozícia. Do zväčšovacieho rámu sa založí fotocitlivý papier. Po stlačení klávesu "e" sa zapne zväčšovací prístroj na dobu, ktorá zodpovedá naposledy zadanej alebo vypočítanej dobe expozície. Opätovným stlačením klávesu možno zopakovať expozíciu toho istého snímku niekoľko krát.

"1/0"– zapnutie a vypnutie zväčšovacieho prístroja prostredníctvom klávesnice. Túto možnosť volí užívateľ pri zaostrovaní snímku, hľadaní výrezu, atď.

Teraz sa vráťme k expozičnému testu. Je nevyhnutný, lebo do určovania doby expozície treba zahrnúť aj vlastnosti vývojky ako i emulzie používaného fotopapiera. K expo-



Obr. 2. Obrazec plošných spojov na doske Y506 (911–2)



Obr. 3. Rozmiestnenie súčiastok na doske Y506 (911–3)

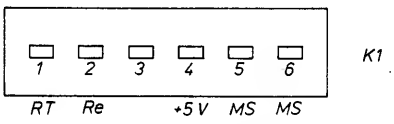
zičnému testu je nevyhnutný negatív stupnice šedi, ktorý sa dá ľahko zhotoviť a to postupným zakrývaním fotopapiera pod zapnutým zväčšovacím prístrojom. Osvedčila sa nám stupnica s 10 prúžkami, pričom krajné prúžky sú úplne čierny a biely. Jej odfotením získame požadovaný negatív stupnice šedi. Predtým než začneme pracovať s AES, zhotovíme pozitív tohoto obrazca na konkrétnom type papiera a vo vývojke, ktoré budeme používať. Dobu expozície obrazca zadáme ako expozičnú konštantu v expozičnom teste. Bez toho, aby sme menili clnové číslo, postupne zmeriame sondou intenzitu všetkých 10 prúžkov, tak ako to pokyny na obrazovke pri expozičnom teste vyžadujú, čím je test ukončený. Ak by napr. vplyvom vyčerpania vývojky dochádzalo k odchýlkam, netreba expozičný test opakovať, ale dá sa zmeniť expozičná konštanta, ako sme sa zmienili už skôr. Celá činnosť s AES je užívateľovi jasná z pokynov zobrazovaných na obrazovke.

Ešte k snímačom. Kondenzátor C1 určuje rozsah snímača intenzity osvetlenia. Jeho presná kapacita je viazaná na určitý typ zväčšovacieho prístroja so svojou svietivosťou, ako i na typ a výrobné odchýlky fotorezistoru. Nelinearitu snímača osvetlenia sme aproximovali 5 úsekmi paraboly a nelinearitu termistora 10 úsekmi paraboly v rozsahu –5 až 100 °C. Červená celofánová fólia drží na obrazovke televízora sama, s využitím jej elektrostatického náboja.

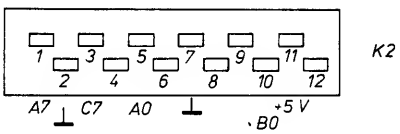
AES sa ukázal byť užitočným pomocníkom šetriacim čas a uľahčujúcim prácu vo fotokomore.

Zoznam použitých súčiastok

IO1,2	E555
IO3	MHB4011
TRK	KT207/600
T1,T2	KC509
T3	KFY16
T4	KFY46
RT	11NR15
RE	WK65061
C1	1 μF (vhodný je tantalový)
C2	680 nF (vhodný je tantalový)



1 – RT
2 – RE
3 –
4 – + 5 V
5 – MS
6 – MS



1 – A7
2 – zem
3 – C7
4 –
5 – A0
6 –
7 – zem
8 –
9 –
10 – B0
11 – + 5 V
12 –

Obr. 4. Zapojenie konektorov (911–4)

C3,C4	47 nF	TK 783
C5	68 nF	TK 783
C6	150 nF	TK 782
C7,C8	200 μF	TE 986
C9	10 nF	TK 783
R1	7,5 kΩ	
R2	12 kΩ	
R3	3,3 kΩ	
R4	22 kΩ	
R5	1,2 kΩ	všetky
R6	11 kΩ	TR 212
R7	4,7 kΩ	
R8	8,2 kΩ	
R9	1 kΩ	
R10	270 Ω	
R11	27 Ω	
MS	mikrospínač (Tesla Jihlava) WN 55900	
Tr	impulzný transformátor o ∅ 20 mm, feritový hrnček typ 505250/H12, primár 20 z ∅ 0,5 mm, sekundár 50 z o ∅ 0,2 mm.	
K1,K2	konektory WK46244, WK46206	

Výpis 1. *Ovládací program v BASICu* (911–V1)

```
500 REM Automatizovany expo-
zicny system
510 REM (c) ing.Ivan HEJDA
512 REM (c) ing.Jan KACMARIK
600 REM Inicializacia
620 POKE 23609,80: CLEAR 49999
625 LET ex=0: PAPER 0: INK 7: B
ORDER 0
630 OUT 127,144: REM Riadiace s
lovo 8255
640 DIM m(4,21): DIM y(21)
650 PRINT AT 13,0;"Automatizova
ny expozicny system"
660 PRINT AT 20,25;"(c) 1986": PA
USE 0
670 CLS : PRINT AT 16,3;"Je ter
mistor ochladovany ?";AT 20,28;"
a/n"
680 IF INKEY$="n" THEN CLS : GO
TO 700
690 IF INKEY$="a" THEN CLS : PR
INT AT 16,5;"Zapni magnetofon":
LOAD "m" DATA m$(): GO TO 700
695 GO TO 680
700 REM Programy pre lokalny
snimac
720 FOR a=50000 TO 50054
730 READ n
740 POKE a,n
750 NEXT a
760 DATA 243,1,0,0,219,31,23,48
,251,3,4,40,8,5,219,31,23,56,246
,251,201,1,0,0,251,201: REM tepl
770 DATA 243,1,0,0,62,0,211,95,
62,128,211,95,3,4,40,8,5,219,31,
31,56,246,251,201,1,0,0,251,201:
REM gfip
790 REM Konstanty pre kvadra-
ticku aproximaciu prevodovej
charakteristiky fotosnimaca
800 DIM a(6): DIM b(6): DIM c(6
)
810 FOR i=1 TO 6
820 READ a(i),b(i),c(i)
830 NEXT i
840 DATA 1.8983e-5,-.027888,11.
2425
850 DATA 7.5156e-7,-.0026034,2.
5
860 DATA 2.86685e-8,-.00025048,
.60945
870 DATA 9.6497e-10,-.000021732
,.13797
880 DATA 3.6945e-11,-2.1576e-6,
.035394
890 DATA 8.2695e-12,-7.6109e-7,
.019243
1000 REM Hlavne menu
1010 CLS : PRINT TAB 13;"M E N U
"
1030 PRINT '"T - Expozicny test
"
1040 PRINT '"K - Korekcia expoz
icnej doby"
1050 PRINT '"E - Expozicny blok
"
1052 PRINT '"F - Meranie teplot
y"
```

```
1054 PRINT "C - Ciachovanie te
rmistora"
1060 PAUSE 0

1070 BEEP 0.02,12
1100 IF INKEY$="e" THEN GO TO 20
00
1110 IF INKEY$="t" THEN GO TO 15
00
1130 IF INKEY$="k" THEN GO TO 17
00
1132 IF INKEY$="f" THEN GO TO 30
00
1134 IF  INKEY$="c" THEN GO TO 3
500
1140 GO TO 1060
1230 PAUSE 0
1490 REM Expozicny test
1500 CLS : PRINT "Expozicny test
"

1505 INPUT "Optimalna doba expoz
icie testo- vacieho snimku ";tte
st
1510 PRINT AT 10,2;"Zosnimaj ska
lu sedi bodovym                sn
imacom"
1520 LET pp=10: REM 10 stupnov
                sedi
1530 DIM e(pp)
1540 OUT 63,1: REM Zapnutie zvec
sovacieho pristroja
1550 FOR i=1 TO pp
1560 PRINT AT 13,12;i;". stlpec"
1570 LET bc=USR 50000: PRINT AT
21,25;bc;"  "
1580 IF bc=0 THEN GO TO 1500
1590 GO SUB 2700: REM Kvadratick
a aproximacia
1600 LET e(i)=jas
1610 NEXT i
1630 GO TO 1000
1690 REM Korekcia expozicnej
         doby
1700 CLS : PRINT "Korekcia expoz
icnej doby"
1710 INPUT "Modifikovana opt. do
ba expozicieskusobneho snimku ";
ttest
1720 GO TO 1000
2000 REM Expozicny blok
2010 REM Menu
2020 CLS : PRINT " E X P O Z I C
 N E     M E N U"
2030 PRINT "Z - zadanie doby e
xpozicie"
2040 PRINT "B - meranie bodovy
m snimacom"
2060 PRINT "E - expozicia (Tex
=";ex;")"
2070 PRINT "1 - zapnutie zvets
. pristroja"
2080 PRINT "0 - vypnutie zvets
. pristroja"
2090 PRINT "M - navrat do menu
"
2100 PAUSE 0
2105 BEEP 0.02,12
2110 IF INKEY$="Z" THEN GO TO 22
00
2120 IF INKEY$="b" THEN GO TO 25
00
2130 IF INKEY$="g" THEN GO TO 26
00
2140 IF INKEY$="1" THEN OUT 63,1
: GO TO 2100
2150 IF INKEY$="0" THEN OUT 63,0
: GO TO 2100
2170 IF INKEY$="e" THEN GO TO 23
00
2180 IF INKEY$="m" THEN GO TO 10
00
2190 GO TO 2100
2199 REM Zadanie doby expozicie
2200 CLS : PRINT "Zadavanie doby
 expozicie"
2210 INPUT "Tex=";ex
2220 GO TO 2000
2300 REM Expozicia
2305 IF ex=0 THEN GO TO 2000
2310 CLS : PRINT "Expozicia"

2320 POKE 23673,0: POKE 23672,0
2330 OUT 63,1
2360 LET ex1=(256*PEEK 23673+PEE
K 23672)/50
2365 IF ex-ex1<0 THEN GO TO 2390
2370 PRINT AT 10,12;ex-ex1;"
"
2380 GO TO 2360
2390 OUT 63,0
2400 GO TO 2000
2499 REM Meranie bodovym      sn
imacom
2500 CLS : PRINT "Meranie bodovy
m snimacom"
2510 INPUT "Zadaj ziadany stupen
 sternania  meraneho miesta ";ss
c
2520 PRINT AT 10,3;"Zvol merane
miesto, aktivuj                sn
imac"
2530 OUT 63,1: REM Zapnutie zvec
sovacieho pristroja
2540 LET bc=USR 50000
2545 OUT 63,0: REM Vypnutie zvec
sovacieho pristroja
2550 IF bc=0 THEN CLS : PRINT FL
ASH 1;AT 10,1;"Intenzita mimo ro
zsahu snimaca": PAUSE 0: BEEP .1
,12: GO TO 2000
2560 GO SUB 2700
2570 LET ex=e(sc)/jas*ttest
2580 GO TO 2000
2700 REM Kvadraticka aproximacia
prevodovej charakteristiky
2710 IF bc<730 THEN LET in=1: GO
 TO 2770
2720 IF bc<1810 THEN LET in=2: G
O TO 2770
2730 IF bc<4290 THEN LET in=3: G
O TO 2770
2740 IF bc<11380 THEN LET in=4:
GO TO 2770
2750 IF bc<29800 THEN LET in=5:
GO TO 2770
2760 LET in=6
2770 LET jas=a(in)*bc*bc+b(in)*b
c+c(in)
2780 RETURN
3000 REM Podprogram pre meranie
          teploty
3005 LET a$="Aktivuj snimac tepl
oty"
3010 CLS : PRINT AT 16,5; FLASH
1;a$
3020 PRINT AT 20,5;"M - navrat d
o menu"
3030 IF INKEY$="m" THEN GO TO 10
00
3040 LET ts=USR 50026
3050 IF ts=0 THEN GO TO 3010
3055 GO SUB 3100: REM Zaradenie
do i-teho intervalu merania.
3060 LET tc=m(2,i)*ts↑2+m(3,i)*t
s+m(4,i)
3070 PRINT AT 16,5;"
"
3080 PRINT AT 10,8;"Teplota=";tc
;"°C";"  "
3090 GO TO 3030
3100 IF ts<m(1,1) OR ts>m(1,21)
THEN PRINT AT 0,0; FLASH 1;"Tepl
ota prekrocila ciachovany
       rozsah! "; GO TO 3120
3110 PRINT AT 0,2;"
 "
3120 IF ts>m(1,11) THEN GO TO 31
90
3130 IF ts>m(1,7) THEN GO TO 317
0
3140 IF ts>=m(1,1) AND ts<=m(1,3
) THEN LET i=1: RETURN
3150 IF ts>m(1,3) AND ts<=m(1,5)
 THEN LET i=2: RETURN
3160 IF ts>m(1,5) AND ts<=m(1,7)
 THEN LET i=3: RETURN
3170 IF ts>m(1,7) AND ts<=m(1,9)
 THEN LET i=4: RETURN
3180 IF ts>m(1,9) AND ts<=m(1,11
) THEN LET i=5: RETURN
3190 IF ts>m(1,15) THEN GO TO 32
20
3200 IF ts>=m(1,11) AND ts<=m(1,
13) THEN LET i=6: RETURN
3210 IF ts>m(1,13) AND ts<=m(1,1
5) THEN LET i=7: RETURN
3220 IF ts>m(1,15) AND ts<=m(1,1
7) THEN LET i=8: RETURN
3230 IF ts>m(1,17) AND ts<=m(1,1
9) THEN LET i=9: RETURN
3240 LET i=10: RETURN
3500 REM Podprogram pre ciacho-
vanie termistora
3502 CLS
3505 PRINT AT 7,6; INVERSE 1;"Ci
achovanie termistora"
3510 FOR i=1 TO 21: PRINT AT 10,
7;"Meranie cislo:";i: INPUT "Tep
lota=";y(i)
3520 LET m(1,i)=USR 50026
3530 NEXT i
3540 CLS : PRINT AT 10,2; FLASH
1;"Prebieha vypocet aproximacie"
3550 FOR i=1 TO 10
3560 LET p=2*i-1: LET d=2*i: LET
 t=2*i+1
3570 GO SUB 3600: REM Riesenie
sustavy lin. rovnic Cramerovym
pravidlom.
3580 NEXT i
3590 SAVE "m" DATA m$( )
3595 GO TO 1000
3600 LET d=m(1,d)*m(1,p)↑2+m(1,t
)*m(1,d)↑2+m(1,p)*m(1,t)↑2-m(1,d
)*m(1,t)↑2-m(1,t)*m(1,p)↑2-m(1,p
)*m(1,d)↑2
3610 LET d1=y(d)*m(1,p)↑2+y(t)*m
(1,d)↑2+y(p)*m(1,t)↑2-y(d)*m(1,t
)↑2-y(t)*m(1,p)↑2-y(p)*m(1,d)↑2
3615 LET d2=y(d)*m(1,p)↑2+y(t)*m
(1,d)↑2+y(p)*m(1,t)↑2-y(d)*m(1,t
)↑2-y(t)*m(1,p)↑2-y(p)*m(1,d)↑2
3620 LET d3=y(t)*m(1,d)*m(1,p)↑2
+y(d)*m(1,p)*m(1,t)↑2+y(p)*m(1,t
)*m(1,d)↑2-y(p)*m(1,d)*m(1,t)↑2-
y(d)*m(1,t)*m(1,p)↑2-y(t)*m(1,p)
*m(1,d)↑2
3630 LET m(2,i)=d1/d
3640 LET m(3,i)=d2/d
3650 LET m(4,i)=d3/d
3660 RETURN
```

Výpis 2. *Rutina pre obsluhu snímačov a prevodníka* (911–V2)

```
 10 ; Rutina pre obsluhu
 20 ; snimacov a prevodnika.
 30
 40          ORG  50000
 45
 50  DI
 60  LD BC,#0000
 70 LC354 IN A,(#1F)
 80  RLA
 90  JR NC,LC354
100 LC359 INC BC
110  INC B
120  JR Z,LC365
130  DEC B
140  IN A,(#1F)
150  RLA
160  JR C,LC359
170  EI
180  RET
190 LC365 LD BC,#0000
200  EI
210  RET
220  DI
230  LD BC,#0000
240  LD A,#00
250  OUT (#5F),A
260  LD A,#80
270  OUT (#5F),A
280 LC376 INC BC
290  INC B
300  JR Z,LC382
310  DEC B
320  IN A,(#1F)
330  RRA
340  JR C,LC376
350  EI
360  RET
370 LC382 LD BC,#0000
380  EI
390  RET
```

MINIGRAF – ZX SPECTRUM

Ing. M. Šedivý, *Hradešínská 25, 101 00 Praha 10*

Článek popisuje využití souřadnicového zapisovače MINIGRAF ve spojení s počítačem ZX Spectrum. Minigraf je výstupní grafická jednotka, pracující v soustavě pravoúhlých souřadnic nezávisle ovládaných vnějším signálem. Použitím MINIGRAFu získáme periférii, vhodnou např. ke kreslení grafů, tabulek, případně obrázků. MINIGRAF však nenahrazuje tiskárnu, především z důvodu malé rychlosti kreslení. Při současném nedostatku podobných zařízení na našem trhu může přinést použití MINIGRAFu určité oživení vašeho malého výpočetního systému.

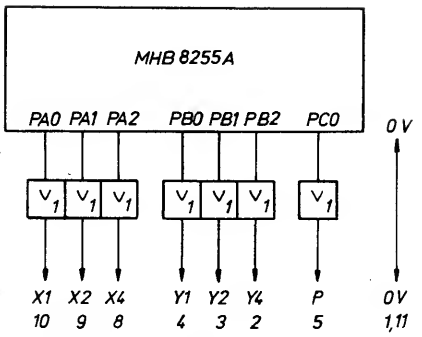
Souřadnicový zapisovač MINIGRAF kreslí v pravoúhlých souřadnicích na běžný kancelářský papír formátu A4, případně na svitek maximální šíře 210 mm. Pisátko je kuličkový fix nebo jiné běžně dostupné pero. Posuv pisátka ve směru osy X a posuv papíru ve směru osy Y zajišťují krokové motorky. Maximální rychlost kreslení je 80 mm/s, délka kroku je 0,125 mm. MINIGRAF je ovládán přes jednoduché jednosměrné rozhraní. Dvěma trojicemi vodičů, označených X1, X2, X4 a Y1, Y2, Y4 jsou ovládány fáze krokových motorků. Sedmý vodič ovládá spouštění pisátka. Pohyb motorků a směr pohybu je řízen rostoucí nebo klesající binární posloupností příslušné trojice bitů. MINIGRAF nedává žádné zpětné hlášení, proto je nutno zajistit časové prodlevy mezi změ-

KROK	VSTUP		
	X4(Y4)	X2(Y2)	X1(Y1)
0	L	L	L
1	L	L	H
2	L	H	L
3	L	H	H
4	H	L	L
5	H	L	H
6	H	H	L
7	H	H	H
8	L	L	L
9	L	L	H
10	L	H	L
11	L	H	H
12	H	L	L
13	H	L	H
14	H	H	L
15	H	H	H
16	L	L	L

Tab. 1. *Posloupnost kroků* (904–T1)

nami binární kombinace programově. Posloupnost jednotlivých kroků je patrna z tabulky č. 1. Při rostoucí posloupnosti (např. kroky 7,8,9) se pohybuje pisátko i papír v kladném směru os (tj. zleva doprava a zezadu dopředu). Jeden krok znamená posun papíru nebo pisátka o 0,125 mm. Maximální kmitočet krokování je 400 Hz. Při reverzaci je nutná prodleva 10 ms a po sklopení nebo zdvižení pisátka je nutná prodleva 50 ms. Pisátko je spouštěno úrovní log.1.

Z uvedeného vyplývá, že pro připojení MINIGRAFu k počítači musíme použít jednu osmibitovou bránu (jeden bit zůstane nevyužit). Pro připojení MINIGRAFu k počítači ZX Spectrum můžeme s výhodou použít interfejs s MHB8255A podle AR č. 6, 1985, s. 217.



Obr. 1. Propojení interfejsu s Minigrafem (na místě oddělovacích zesilovačů lze použít např. 2× 7407, 1× 3212, 2× 3216, 1× 8282 ap.) (904–1)

Při propojení jsem pro jednodušší programové řešení využil všechny tři brány obvodu 8255. Každý ovládací bit má v MINIGRAFu udávanou log. zátěž 5, proto je vhodné všechny signály zesílit. Propojení interfejsu s MINIGRAFem je na **obr. 1.** Po propojení a zadání následujícího jednoduchého programu by se měl motor pro posun papíru točit.

```
10 OUT127,128
20 FOR A=0 TO 15
30 OUT 63,A
40 NEXT A
50 GOTO 20
```

Zaměníme-li výstup 63 za 31, měl by se točit motor pro posun pisátka. Instrukcí OUT 95, 255 spustíme pisátko. Tím je propojení odzkoušeno a můžeme přistoupit k programovému řešení.

Ovládání MINIGRAFu v jazyce BASIC je pomalé, a proto je vhodné psát programy ve strojovém kódu. Uvádím příklad programu, který provádí kopii obrazovky na souřadnicovém zapisovači. Program je uložen na konci paměti za RAMTOP. Program nahrajeme např. takto: CLEAR 65043 : LOAD " " CODE 65044 : POKE 65372, X : POKE 65414, X. Proměnná X může nabývat hodnot od 1 do 8 a určuje rozměr kopie. Po nahrání je nastaveno X = 4. Program spustíme příkazem RANDOMIZE USR 65044. Po spuštění programu nastavíme ručně pisátko do výchozího bodu při levém okraji na začátku papíru. K ručnímu ovládání posuvu pisátka a papíru slouží tlačítka se šipkami pro ovládání kurzoru.

Kreslení spustíme tlačítkem EDIT, přerušení kreslení a návrat z programu způsobí stisk tlačítka CAPS LOCK. Návrat před spuštěním kreslení je možný tlačítkem DELETE. Po ukončení kresby se automaticky provede návrat z programu.

Pro výpis programu není pro svou pomalost zapisovač vhodný, ale v nouzi poslouží také. Pro kreslení je naopak velmi vhodný, umožňuje jemnější kresbu než obrazovka nebo tiskárna.

Literatura

|1| *Soldán,J.:* Interfejs s MHB8255. AR/A, č. 6,1985, s. 217–219.

|2| MINIGRAF 0507, návod k obsluze, Aritma 1986.

Výpis 1. *Strojový kód programu Zapisovač* (904–V1)

```
65044: 243 245 213  62 128 211 127  17   0   0
65054:  62   0 211  95  62 239 219 254 203 103
65064: 194  44 254  29 203  95 194  50 254  28
65074: 203  87 194  56 254  20 203  71 202  87
65084: 254  62 247 219 254 203 103 194  71 254
65094:  21 203  71 202  84 254 217 205  96 254
65104: 217 195  34 254 205 156 254 209 241 251
65114: 201   0   0   0   0   0 217 122 211  31
65124: 123 211  63 205 112 254 217 201   0   0
65134:   0   0 197   6   2  14 255  13 194 117
65144: 254   5 194 115 254 193 201 205 112 254
65154: 205 112 254 205 112 254 205 112 254 201
65164: 205 127 254 205 127 254 205 127 254 201
65174:   0   0   0   0   0   0 197 213 229 245
65184:  30   3 217 197 229  17   0   0   1   0
65194:   0 217  33   0  64   6   8  14   8  22
65204:  32 217   6   0 217  52  53 196   0 255
65214:  35 217  62   8 128  71 217  21 194 185
65224: 254 197   1 224   0   9 193 217  12 217
65234: 205 127 254  13 194 179 254 197   1  32
65244: 248   9 193   5 194 177 254 197   1   0
65254:   7   9 193  29 194 175 254 217   1   0
65264:   0 217 205   0 255 217 225 193 217 241
65274: 225 209 193 201   0   0 197 213 217 120
65284: 186 212  19 255 210  15 255  21 195 107
65294: 255  20 195 114 255 121 187 202  39 255
65304: 210  31 255  29 195 121 255  28 195 128
65314: 255   0   0   0   0 217   6   8 126  87
65324: 122  23  87 218  77 255  62   0 211  95
65334: 205 140 254 195 200 255 195 160 255   5
65344: 194  44 255  62   0 211  95 205 140 254
65354: 209 193 201  62 255 211  95 205 140 254
65364: 195  57 255   0  07   0 197   6   4 124
65374: 211  63  60 103 205 112 254   5 194  93
65384: 255 193 201 213  17   0 255 195 132 255
65394: 213  17   0   1 195 132 255 213  17 255
65404:   0 195 132 255 213  17   1   0 197   6
65414:   4 124 211  63 130 103 125 211  31 131
65424: 111 205 112 254   5 194 135 255 193 209
65434: 195 183 255   0   0   0 217 205  90 255
65444: 217 195  63 255   0   0   0   0   0   0
65454:   0   0   0   0   0   0   0   0   0  62
65464: 247 219 254 203  79 194   3 255 217 209
65474: 193  51  51 195 245 254 217  20  62   0
76484: 186 194 214 255  22 255 217 195  63 255
65494: 217 195  60 255
```

PROGRAMOVÁNÍ PAMĚTÍ EPROM
na počítačích ATARI XE/XL

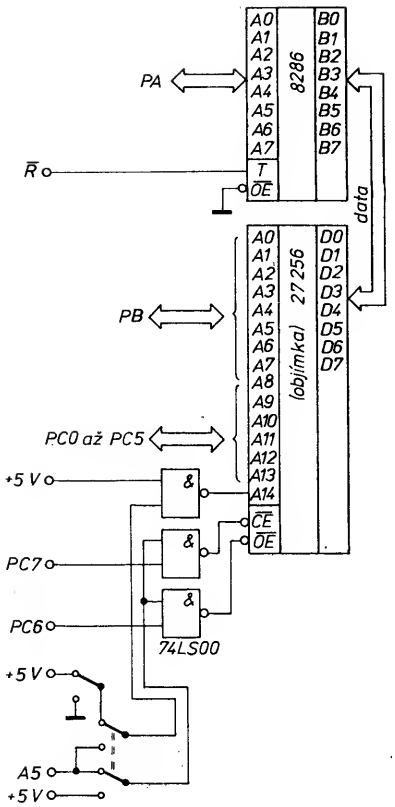
Jiří Pokorný, Jan Pokorný, Jiří Pokorný, *Fr. Kadlece 18, 180 00 Praha 8*

Standardní programátory pamětí EPROM nejsou pro většinu amatérů dostupné. Pro malé paměti se nejčastěji amatérsky vyrábějí „ruční" programátory, pro větší paměti je nutno programovat pomocí řídícího mikropočítače. Uvedeme popis programování pamětí EPROM 27128 (16 kB) a 27256 (32 kB) s využitím počítače ATARI řady XL nebo XE. Popsaný postup byl použit při ověření funkce programového modulu (cartridge) s jazyky BASIC XE – MAC/65.

Technické řešení

Adresy, data i řídící signály se přivádějí na paměť EPROM přes paralelní interfejs s MHB 8255A, jehož popis a schéma jsou uvedeny v |1|. Paralelní interfejs je zapojen k počítači ATARI v konektoru pro programový modul a na joystickové porty. K paralelnímu interfejsu je připojen přípravek s objímkou pro paměť EPROM. Schéma zapojení přípravku je na **obr. 1.** Adresy programované EPROM A0 až A7 jsou připojeny na port PB obvodu 8255 A, adresy A8 až A13 na část portu PC. Při programování EPROM 32 kB je přes přepínače a invertor přiváděna na vývod číslo 27 (kde je adresa A14) buď úroveň log.0 (programování v rozmezí $0 až $3FFF) nebo úroveň log.1 (programování v rozmezí $4000 až $7FFF). Při programování EPROM 16 kB se tento vývod používá jako PGMn. Data jsou vyvedena z portu PA na oddělovací obvod MHB 8286. Směr průchodu dat tímto obvodem lze ovládat signálem Rn obvodu 8255A nebo signálem OEn programované EPROM. Řídící vstupy EPROM CEn, OEn jsou ovládány bity PC7, PC6. Při programování EPROM 32 kB jsou tyto signály kombinovány s latchovanou adresou A5, která působí jako hradlový signál. Při programování EPROM 16 kB latchovaná adresa A5 vytváří programovací impuls pro PGMn. V zapojení nejsou zakresleny přívody napájecích napětí U_{CC}, U_{pp} a příslušné blokovací kondenzátory (paralelně 100 nF a 5μF). Provozní hodnoty U_{CC} i U_{pp} jsou +5 V. Při programování je U_{CC} = = 6 V a U_{pp} = 12,5 V. U některých EPROM 27128 je U_{pp} = 21 V (neuvádíme rozmezí napětí).

Obvod 74LS00 se používá jako invertor a oddělovací stupeň. Při programování EPROM 27256 působí rovněž jako hradlo pro signály řízené CEn a OEn.



Obr. 1. Schéma přípravku k programování EPROM 27256 a 27128 (915–1)

Programové řešení

Algoritmy programování pamětí EPROM včetně vývojových diagramů jsou uvedeny v katalogu firmy Intel. Používáme algoritmus nazývaný „inteligent". Programuje se milisekundovými impulsy; po každém z nich následuje verifikace zapsaného bajtu. Pokud byl bajt zapsán správně, programuje se závěrečným impulsem („overimpulsem"). Délka tohoto impulsu je trojnásobkem celkové délky všech milisekundových impulsů potřebných k zapsání tohoto bajtu.

Jestliže se programovaný bajt nezapsal ani po 25 impulsech délky 1 ms, je ohlášena chyba. Po zapsání všech bajtů programu do paměti EPROM se provede verifikace celého programovaného úseku při provozních hodnotách napětí.

Hlavní program je vypracován v ASSEMBLERU CPU 6502 a je uložen od adresy $8000. Je volán obslužným programem v BASIC. Programy jsou uvedeny ve **Výpisu 1** a **Výpisu 2.**

Ovládání obou programů

RUN + RETURN

Start programu. Na obrazovce se zobrazí text:

„Programovani 16/32 kB EPROM (zadej 16 nebo 32) . . . ?"

Po zadání požadované hodnoty a stisku klávesy RETURN se na displeji zobrazí text *„Adresa EPROM, od ktere se zacina programovat . . . ?".*

Na tuto výzvu uložíme příslušnou adresu (decimálně) a stiskneme RETURN. Na displeji uvidíme

„Adresa RAM, kde zacinaji data pro programovani EPROM . . . ?".

Zadáme adresu začátku dat (decimálně) a stiskneme RETURN.

Na displeji se zobrazí další text:

„Adresa RAM, kde konci data pro programovani . . . ?".

Zadáme koncovou adresu RAM, kde jsou uložena data. Program pokračuje po stisku RETURN. Na displeji se zobrazí

„Kontroluj napeti Vcc a Vpp (+5 V) na pameti EPROM! (Cekam na klavesu!)".

Po stisku libovolné klávesy tento program v BASIC vyvolá program ve strojovém kódu pro programování EPROM. Na displeji uvidíte

„Nastav napeti Vcc = 6 V, Vpp = 12,5 V. (Cekam na klavesu! – A, B)"

Na tuto výzvu nastavíte programovací napětí v uvedeném pořadí na paměti EPROM (podle popisu v katalogu Intel). Pro některé EPROM 27128 je třeba V_{pp} = 21 V. Po stisknutí kláves A a B v uvedeném pořadí program provádí potřebné programovací opera-

Výpis 2. *Program ve strojovém kódu k programování paměti EPROM (ukládá se od adresy $8000)* (915–V2)

```
68 20 02 81 AD 11 06 F0 06 20 12 81 4C 12 80 20 0A 81 20  0521
EB 80 20 72 81 20 90 81 20 11 83 A0 00 B1 CB 20 74 83 20  07B6
25 81 A9 00 8D 00 D4 A5 CB 8D 0B 06 A5 CC 8D 0C 06 A2 00  0770
A0 00 B1 CB 8D 08 06 A0 01 20 9F 81 E8 E0 19 D0 03 4C CB  0863
80 20 39 82 AD 08 06 CD 09 06 D0 E8 8A 8E 0A 06 18 6D 0A  0661
06 6D 0A 06 A8 20 9F 81 EE 06 06 D0 03 EE 07 06 E6 CB D0  07B4
02 E6 CC A5 CB CD 04 06 D0 B1 A5 CC CD 05 06 D0 AA AD 2F  0A1B
02 8D 00 D4 20 1A 81 A9 00 8D 00 D4 20 EB 80 A0 00 20 C3  0736
82 A0 00 B1 CB CD 09 06 D0 37 E6 CB D0 02 E6 CC EE 06 06  09B0
D0 03 EE 07 06 A5 CB CD 04 06 D0 DF A5 CC CD 05 06 D0 D8  09B5
AD 2F 02 8D 00 D4 20 84 81 20 3B 81 60 20 39 82 AD 08 06  0636
CD 09 06 D0 03 4C 58 80 20 84 81 AD 2F 02 8D 00 D4 20 4C  06A3
83 20 30 81 4C CA 80 AD 00 06 8D 06 06 AD 01 06 8D 07 06  0584
AD 02 06 85 CB AD 03 06 85 CC 60 A2 C1 A0 84 20 42 C6 60  087B
A2 05 A0 85 20 42 C6 60 A2 35 A0 85 20 42 C6 60 A2 34 A0  084E
84 20 42 C6 20 46 81 60 A2 A7 A0 83 20 42 C6 20 46 81 60  07CE
A2 ED A0 83 20 42 C6 20 46 81 60 A2 7A A0 84 20 42 C6 20  08A9
46 81 60 AD 09 D2 C9 3F D0 F9 AD 09 D2 C9 15 D0 F9 60 8A  0A99
48 88 F0 0C A2 B2 EA EA EA CA D0 FA A2 00 F0 F1 A2 AE EA  0D2F
EA EA CA D0 FA 68 AA EA 60 08 68 8D 0E 06 78 AD 0E D4 8D  0A69
0D 06 A9 00 8D 0E D4 60 AD 0D 06 8D 0E D4 AD 0E 06 48 28  05EB
60 8D E3 D5 20 2C 83 A9 80 20 84 83 8D E0 D5 60 AD 11 06  092A
F0 6C 20 11 83 AD 08 06 20 74 83 8D E3 D5 A9 0F 20 84 83  0806
8D D0 D5 20 55 81 8D F0 D5 A9 0E 20 84 83 8D E0 D5 60 48  0A42
C0 01 D0 0D A9 03 8D 0F 06 A9 01 8D 10 06 4C E3 81 A9 0A  069C
8D 0F 06 A9 01 8D 10 06 EE 10 06 CE 10 06 EE 10 06 CE 10  05B9
06 EE 10 06 CE 10 06 EE 10 06 CE 10 06 CE 0F 06 D0 E3 A9  0715
FF 8D 0F 06 CE 10 06 D0 D9 EA EA EA EA 68 60 20 11 83 AD  09FF
08 06 20 74 83 8D E3 D5 A9 0F 20 84 83 20 C7 81 8D D0 D5  08E3
20 55 81 8D F0 D5 20 C7 81 A9 0E 20 84 83 8D E0 D5 60 AD  09DD
11 06 D0 03 4C 7D 82 A9 90 20 84 83 20 11 83 A9 0D 20 84  06A3
83 20 3C 83 8D F3 D5 8D F7 D5 8D D7 D5 20 C7 81 AD 00 D3  0B31
8D 09 06 8D F7 D5 8D F3 D5 8D E3 D5 20 2C 83 A9 0C 20 84  09B7
83 A9 80 20 84 83 8D E0 D5 60 A9 90 20 84 83 20 11 83 A9  0932
0F 20 84 83 20 C7 81 A9 0D 20 84 83 20 3C 83 8D F3 D5 8D  083C
F7 D5 20 C7 81 AD 00 D3 8D 09 06 8D F3 D5 8D E3 D5 20 2C  0A36
83 A9 0C 20 84 83 20 C7 81 A9 0E 20 84 83 A9 80 20 84 83  07F5
8D E0 D5 60 AD 11 06 D0 03 4C 7D 82 A9 90 20 84 83 20 11  0815
83 A9 0D 20 84 83 A9 0F 20 84 83 20 3C 83 8D F3 D5 8D F7  08F7
D5 8D D7 D5 20 C7 81 AD 00 D3 8D 09 06 8D F7 D5 8D F3 D5  0B40
8D E3 D5 20 2C 83 A9 0E 20 84 83 A9 0C 20 84 83 A9 80 20  0817
84 83 8D E0 D5 60 AD 06 06 8D 00 D3 20 C7 81 8D F2 D5 8D  0A0B
FA D5 8D F2 D5 AD 07 06 29 3F 20 94 83 60 A9 38 8D 02 D3  091F
A9 FF 8D 00 D3 A9 3C 8D 02 D3 60 A9 38 8D 02 D3 A9 00 8D  0928
00 D3 A9 3C 8D 02 D3 60 AD 0B 06 85 D4 AD 0C 06 85 D5 20  07CA
AA D9 20 E6 D8 A0 FF C8 B1 F3 10 FB 29 7F 91 F3 C8 A9 9B  0CAF
91 F3 A6 F3 A4 F4 20 42 C6 60 8D 00 D3 20 C7 81 8D F3 D5  0B5A
8D FB D5 8D F3 D5 60 8D 00 D3 20 C7 81 8D F0 D5 8D F8 D5  0C86
8D F0 D5 60 8D 00 D3 20 C7 81 8D F1 D5 8D F9 D5 8D F1 D5  0C7B
8D E1 D5 60 20 20 20 4E 61 73 74 61 76 20 6E 61 70 65 74  07A8
69 20 56 63 63 3D 36 56 2C 20 56 70 70 3D 31 32 2E 35 56  0549
2E 20 20 20 20 20 20 20 20 20 28 43 65 6B 61 6D 20 6E  0405
61 20 6B 6C 61 76 65 73 75 21 20 2D 41 2C 42 29 9B 1D 20  059A
20 20 45 50 52 4F 4D 20 6E 65 6C 7A 65 20 6E 61 68 72 61  062B
74 20 28 58 3D 32 35 29 2E 20 20 20 20 20 20 20 20 20 20  034F
20 20 20 20 20 20 28 43 65 6B 61 6D 20 6E 61 20 6B 6C 61  0510
76 65 73 75 21 20 2D 41 2C 42 29 9B 20 20 20 53 6E 69 7A  05A8
20 6E 61 70 65 74 69 20 56 70 70 3D 35 56 2C 20 56 63 63  0627
3D 35 56 2E 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20  02D6
28 43 65 6B 61 6D 20 6E 61 20 6B 6C 61 76 65 73 75 21 20  0654
2D 41 2C 42 29 9B 1D 20 20 20 45 50 52 4F 4D 20 6E 61 70  04FF
72 6F 67 72 61 6D 6F 76 61 6E 61 21 20 20 20 20 20 20 20  059E
20 20 20 20 20 20 20 20 20 20 20 20 20 20 28 43 65 6B 61  035C
6D 20 6E 61 20 6B 6C 61 76 65 73 75 21 20 2D 41 2C 42 29  05BD
9B 7D A0 A0 D0 D2 CF C7 D2 C1 CD CF D6 C1 CE C9 A0 C5 D0  0E22
D2 CF CD A0 B2 B7 B1 B2 B8 A0 E1 A0 B2 B7 B2 B5 B6 A0 A0  0D79
20 20 20 20 20 20 20 20 A0 A0 A0 C8 F5 ED E2 EC E5 A0 D3  09B0
EF E6 F4 F7 E1 F2 E5 A0 A0 A0 1D 9B 20 20 50 72 6F 67 72  0B5A
61 6D 6F 76 61 6E 69 20 45 50 52 4F 4D 20 32 37 31 32 38  05B2
20 28 31 36 6B 42 29 21 20 20 20 20 1D 1D 50 6F 6B 79 6E  0471
79 3A 9B 20 20 50 72 6F 67 72 61 6D 6F 76 61 6E 69 20 45  06E8
50 52 4F 4D 20 32 37 32 35 36 20 28 33 32 6B 42 29 21 20  0428
20 20 20 1D 1D 50 6F 6B 79 6E 79 3A 9B                    03F9
```

```
10 REM ****************************
20 REM *   OBSLUZNY PROGRAM PRO    *
30 REM *      NAHRAVANI EPROM      *
40 REM * 11.7.88. Humble Software  *
50 REM ****************************
60 ? "Programovani 16/32 kB EPROM (zadej 16 nebo 32) ... ";
70 INPUT DRUH
80 IF DRUH<>16 AND DRUH<>32 THEN 60
90 DRUH=DRUH/16-1
100 ? "Adresa EPROM, od ktere se zacina prog-ramovat ...";
110 INPUT ZACEP
120 ? "Adresa RAM,kde zacinaji data pro prog-ramovani EPROM ..
.";
130 INPUT ZACRAM
140 ? "Adresa RAM, kde konci data pro progra-movani ...";
150 INPUT KONRAM
160 KONRAM=KONRAM+1
170 POKE 1536,ZACEP-INT(ZACEP/256)*256
180 POKE 1537,INT(ZACEP/256)
190 POKE 1538,ZACRAM-INT(ZACRAM/256)*256
200 POKE 1539,INT(ZACRAM/256)
210 POKE 1540,KONRAM-INT(KONRAM/256)*256
220 POKE 1541,INT(KONRAM/256)
230 POKE 1553,DRUH
240 ? "Kontroluj napeti Vcc a Vpp (+5V) na   pameti EPROM!
                    (Cekam na klavesu!)"
250 POKE 764,255
260 IF PEEK(764)=255 THEN 260
270 C=USR(32768)
280 ? "   KONEC!"
290 END
30000 POKE 54018,56:POKE 54016,255:POKE 54018,60
30010 POKE 54016,128
30020 POKE 54768,0
30030 POKE 54776,0
30040 POKE 54768,0
30050 POKE 54016,0
30060 POKE 54771,0
30070 POKE 54779,0
30080 POKE 54771,0
30090 POKE 54770,0
30100 POKE 54778,0
30110 POKE 54770,0
30115 POKE 54016,128+64
30120 POKE 54769,0
30130 POKE 54777,0
30140 POKE 54769,0
```

Výpis 1. *Základní program v jazyku BASIC (915–V1)*

ce. Po úspěšném naprogramování paměti EPROM se na displeji zobrazí nápis *„Sniz napeti Vpp=5 V, Vcc=5 V. (Cekam na klavesu! – A, B)“.*
Po snížení programovacích napětí (v uvedeném pořadí) na určené hodnoty a stisknutí kláves A a B program provádí verifikaci naprogramovaných bajtů. Pokud nebyla nalezena chyba, zobrazí se na displeji nápis *„EPROM naprogramovana. (Cekam na klavesu! – A, B)“*
Po stisknutí uvedených kláves následuje návrat do programu v BASIC a nápis *„KONEC!“.*

Tímto postupem je celá požadovaná oblast paměti EPROM naprogramována.

V případě, že ani po 25 jednomilisekundových impulsech není do paměti EPROM správně zapsán požadovaný bajt, objeví se nápis
„EPROM nelze nahrat (X=25). (Cekam na klavesu! – A, B)“.
Po stisku kláves A, B se vrátíte do programu v BASIC a zobrazí se nápis
„KONEC!“.

V případě, že při verifikaci po skončení programování při provozních hodnotách napětí byl v paměti EPROM nalezen bajt, který neodpovídá příslušnému bajtu v RAM počítače, na obrazovce se objeví stejný text, jako při chybném zápisu po 25 impulsech délky 1 ms a po stisku kláves A, B se program vrátí do BASIC a zobrazí se nápis
„KONEC!“.

Před zasunutím a vyjmutím EPROM z objímky přípravku se nastaví logická 0 na přívodech EPROM příkazem

GOTO 30000 + RETURN.

Závěr

Zvolené hardwarové uspořádání umožňuje programovat EPROM 27128 (16 kB) nebo 27256 (32 kB). V případě EPROM 27256 se nejdříve programuje 16 kB a po změně úrovně na adrese A14 dalších 16 kB.
Programování paměti EPROM jsme ověřili s paralelním interfejsem využívajícím joystickové porty a port pro programový modul (popsaný v [1]). Zápis 16 kB včetně verifikace trvá asi 4 min.

Literatúra

[1] *Pokorný J. et al.:* Univerzální paralelní interfejs k počítačům ATARI 800XL a 130XE. Zasláno do Amatérského Radia.

PROGRAMÁTOR EPROM 8708
ŘÍZENÝ ŠKOLNÍM MIKROPOČÍTAČEM PMI-80

Ing. Jaroslav Pipek, *Horská 439, 543 02 Vrchlabí*

Článek se zabývá popisem programovacího přípravku přímo slučitelného se školním mikropočítačem PMI 80. Obvodové řešení však může být vodítkem i pro vlastníky jiných mikropočítačů, kteří nemají možnost paměti EPROM programovat. Přestože PMI 80 patří mezi nejjednodušší, lze jej vedle praktického procvičování instrukčního souboru mikroprocesoru MHB8080 úspěšně využít jako základní jednotku pro realizaci celé řady i poměrně složitých automatizačních a regulačních zařízení. Je ovšem nutné tuto základní jednotku patřičně rozšířit jak obvodově, tak programově v souladu s danou aplikací.

Při návrhu a oživování zařízení se neobejdeme (alespoň v počátečním stadiu) bez použití paměti typu EPROM, které umožňují trvale uchovat informaci bez ohledu na jejich zapojení do obvodu. Lze je však vymazat ozářením ultrafialovými paprsky. To umožňuje použití těchto pamětí i všude tam, kde již program je funkční, ale kde se dá předpokládat, že časem dojde k jeho obměně.

Navržený přípravek umožní uživatelům PMI 80 pohotově kdykoli změnit obsah těchto pamětí nezávisle na jiném systému.

Popis přípravku

Přípravek je osazen pouze dvěma IO (MH7400 a UCY74123) a šesti tranzistory. Odpadá adresový i datový registr. Jako registrů je využito údajů na datové a adresní sběrnici ve stavu WAIT mikroprocesoru. Celý přípravek je sestaven na oboustranně plátované kuprextitové destičce o rozměrech 81×80 mm a zasouvá se do konektoru K1 na desce PMI-80.

Obsluha a činnost

Nejprve jsou data určená k zápisu do EPROM zapsána z klávesnice PMI-80 do zóny RAM paměti (zvolena oblast 1C00 až 1CFF). Tento blok dat lze přepsat ze zóny RAM do libovolné oblasti programované paměti EPROM v rozsahu 0000 až 03FF. Počáteční adresu této oblasti předem uložíme z klávesnice do pomocných buněk 1F00 a 1F01 v oblasti paměti RAM PMI-80. Přitom do buňky 1F00 ukládáme nižší bajt této adresy v hexadecimálním tvaru, vyšší bajt do buňky 1F01. Údaje vyššího bajtu však korigujeme vždy o + 20H (viz **tab.1**) s ohledem na aktivaci programované EPROM signálem A13. Do další pomocné buňky 1F02 ukládáme v hexadecimálním tvaru údaj o délce bloku dat v RAM (jinak řečeno obsah 1F02 udává nižší bajt koncové adresy bloku dat v RAM). S ohledem na skutečnost, že jsme pro data určená k přepisu do EPROM vymezili v RAM oblast 1C00 až 1CFF, je délka bloku dat max. 256 bajtů. Jednodušeji lze však naprogramovat až 1 kB dat, bude-li se opakovat programovací cyklus (s novou

náplní buněk 1F00 až 1F02 podle **tab.1** a s novým blokem dat umístěným opět do zóny 1C00 až 1CFF celkem čtyřikrát. Je to nutné vzhledem k nedostupnosti dalšího 1 kB paměti RAM. Bude-li k dispozici celý 1 kB RAM pouze pro účel programování, pak lze programovat EPROM najednou (ovšem s patřičně změněným řídicím programem). Řídicí program je umístěn v pomocné paměti EPROM, která je zasunuta do volné pozice na desce PMI-80 (její počáteční adresa je 0400). Programovaná paměť se umisťuje na desku programátoru. Programování (přepis z RAM) začíná spuštěním řídicího programu „PROGRAM" na adrese 0400 (řídicí program lze umístit libovolně, podle toho se mění startovací adresa). Data určená pro zápis do EPROM jsou postupně načítána ze zóny paměti RAM (1C00 až 1CFF) a přesunovou instrukcí MOV M, A prostřednictvím bitu A13 adresní sběrnice je startován monostabilní klopný obvod MKO1, který vytváří signál READY a uvede mikroprocesor do stavu WAIT. Současně je spínacím obvodem, tvořeným tranzistory T5 a T6, přivedena na vstup CSn/WE úroveň 12 V. Stav adresní sběrnice se nemění, data na datové sběrnici jsou připravena od okamžiku přechodu signálu WR na log.0. Současně s přechodem signálu WAIT na úroveň H je přes zpožďovací obvod, tvořený dvěma hradly MH7400, spuštěn druhý monostabilní klopný obvod MKO2, který generuje impuls pro připojení programovacího napětí U_{prog} spínačem s tranzistory T1 a T2. Je-li programování ukončeno, na displeji se rozsvítí nápis END. Tlačítkem RESET přejdeme do výchozího stavu mikroprocesoru, doprovázeného nápisem PMI-80.

Správnost zapsaných dat mohu okamžitě překontrolovat spuštěním programu „KOMPARACE" na adrese 0430. Je-li některý bajt zapsán chybně, zobrazí se v datovém poli displeje číslo tohoto bajtu. Byl-li celý blok naprogramován správně, rozsvítí se nápis „C dobrE". Obsah programované paměti lze též kdykoli přečíst běžným způsobem počínaje adresou 2C00. K tomuto účelu slouží spínač z tranzistorů T3 a T4.

Další možnosti řídicího programu

Často potřebujeme překopírovat obsah jedné paměti EPROM do druhé. V tomto případě si pomůžeme následovně, aniž bychom museli vyvádět sběrnice. První paměť (ORIGINÁL) umístíme do objímky, připájené shora k vývodům na paměti MHB8608 s programem MONITOR. Pouze vývod CSn je propojen lankem s pozicí CS4n na konektoru K1. Druhá EPROM (KOPIE), na níž bude pořizována kopie, je v objímce na desce programátoru.

Spuštěním uživatelského programu KOPIE na adrese 0410 se přepíše celá paměť z pozice ORIGINÁL na pozici KOPIE (program testuje stavy FF, které přeskakuje). V tomto případě může být obsah pomocných buněk 1F00 až 1F02 libovolný. Program tyto buňky nerespektuje.

V případě, že potřebujeme změnit obsah dat v EPROM, umístíme tuto do pozice ORIGINÁL. Na pozici kopie umístíme paměť prázdnou. Spuštěním programu PREPIS na adrese 0470 (v souladu s **tab. 1**) se přepíše požadovaný blok dat do oblasti RAM na adresy 1C00 až 1CFF, kde je možné běžným způsobem (viz obsluha PMI 80) změnit obsah požadovaných buněk. Blok dat po opravě přepíšeme do EPROM spuštěním již známého programu PROGRAM na adrese 0400. Tuto možnost přechodu přes RAM lze využít i pro přemístění bloku dat z dané pozice původní EPROM do libovolné oblasti nové EPROM.

Tab. 1. Pomocné buňky v paměti RAM (914–T1)

1F00	**obsahuje nižší bajt adresy, udávající počátek bloku dat v EPROM, a to:** **a) v programované EPROM při programu "PROGRAM",** **b) ve snímané EPROM při programu "PŘEPIS".**
1F01	**obsahuje vyšší bajt, závisí na oblasti EPROM, kterou programuji při programu PROGRAM, nebo z které snímám při programu PŘEPIS.**

programovaná oblast přepisovaná oblast EPROM	obsah 1F01
0000 - 00FF	20
0100 - 01FF	21
0200 - 02FF	22
0300 - 03FF	23

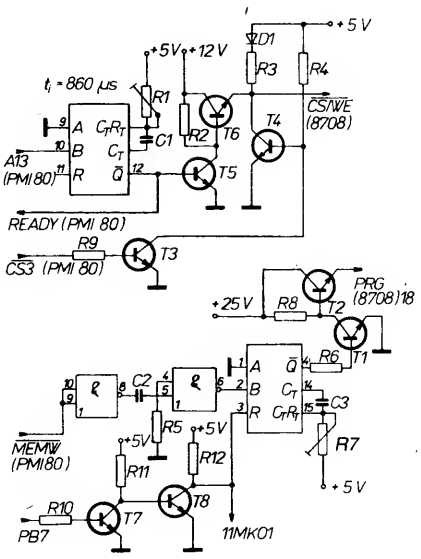
Poznámka

Jelikož není použit adresový dekodér, ale přímo bit A13 na přechod do stavu WAIT, mohlo by v důsledku činnosti programu MONITOR dojít k přepsání již naprogramovaných správných dat při ponechání programovacího napětí + 25 V. Proto je využit výstup PB7 IO10 pro blokování obou MKO mimo vlastní program „PROGRAM". Oba jsou blokovány vždy od okamžiku připojení zdroje k PMI-80 a po stisku tlačítka RESET (neboť bit PB7 brány PB IO10 je ve třetím stavu, tudíž T8 je sepnut). Programově je zajištěno, že pouze při programování přechází výstup PB7 do stavu H a tím uvolňuje činnost obou MKO (UCY74123). Vývod PB7 je z pozice 41 konektoru K2 spojen lankem na vstup T7. Signál z kolektoru T8 je přivede na nevyužitou pozici 28 konektoru K1. Obvod s tranzistory T7 a T8 je umístěn v univerzálním poli přímo na desce PMI-80.

Oživení a nastavení

Zapojení je natolik jednoduché, že s oživováním přípravku nebudou problémy. Po osazení a překontrolování vodivých cest podle **obr. 1** nastavíme běžce trimrů asi do poloviny dráhy.

První známka správné funkce bude, když po zasunutí přípravku do konektoru K1 (při přerušeném vývodu z kolektoru T8 na nulovací vstupy UCY74123) bude blikat nápis „PMI-80". Podle | 1 | by měla být délka programovacího pulsu t_{PW} v rozmezí 0,1 až 1 ms. Pro spolehlivé programování paměti musí být současně splněna nerovnost $N.t_{pw} \geq 100$ ms, kde N je počet programovacích cyklů.



Obr. 1. Schéma přípravku (914–1)

Volíme N = 128 (viz příkaz **MVI B,80** na adrese 00FB), t = 800 μs. Pomocí osciloskopu nastavíme trimrem R7 dobu kyvu MKO 2 asi na 800 μs, trimrem R1 dobu kyvu MKO1 asi na 860 μs. Zapojení s těmito časy funguje spolehlivě. Dále připojíme programovací napětí. Na výstupu emitoru T2 se musí objevit pulsy s úrovní + 25 V a délkou 800 μs. Po připojení vývodu z kolektoru T8 na nulovací vstupy UCY74123 je přípravek připraven k činnosti. Zbývá ještě uložit do pomocné paměti EPROM vlastní řídicí program podle **Výpisu 2.** Jedna z možností, jak tuto paměť již v tomto stadiu naprogramovat bez použití jiného programátoru, je umístit řídicí program „PROGRAM" (zahrnuje části začínající adresami: PROGRAM, PRESUN, ZOBRAZ) např. počínaje adresou 1D00 do paměti RAM. Nyní můžeme již známým způsobem uložit celý řídicí program podle **Výpisu 2** s tím, že startovací adresa je 1D00 (pozor též na změnu adres ve skokových instrukcích).

Vlastní programování si názorně objasníme na dvou příkladech. Představme si nejprve, že je třeba do paměti EPROM počínaje adresou např. 01AA zapsat následující blok dat: F3, 3E, 20, 06, 80, 21, FF, 1E, 23, 00, 76. Postupně provedeme následující kroky:

1) Data zapíšeme do paměti RAM počínaje adresou 1C00:

1C00 F3
1C01 3E
1C02 20
1C03 06
1C04 80
1C05 21
1C06 FF
1C07 1E
1C08 23
1C09 00
1C0A 76

Délka bloku dat, jak je zřejmé, je 0A.

2) V souladu s **tab. 1** zapíšeme na pomocné buňky:

1F00 AA (nižší bajt počáteční ukládací adresy)
1F01 21 (vyšší bajt + 20H)
1F02 0A (délka bloku dat)

3) Odstartujeme program PROGRAM příkazem G.

Výpis 1. *Program v symbolických instrukcích 8080. Startovací adresa = absolutní adresa EPROM + 0400H (paměť je aktivována vstupem CS1n)* (914–V1)

```
(914-V1)
Adresa
absolutní

0000  PROGRAM:    MVI C,FF    ; pomocná data
0002              LHLD 1F00   ; vytvoření počáteční ukládací
                                adresy
0005              LXI D,1C00  ; počátek bloku dat v RAM
0008              JMP PRESUN

0010  KOPIE:      LXI D,1000  ; počátek bloku dat v EPROM
                                "ORIGINAL"
0013              LXI H,2000  ; počáteční adresa v EPROM
                                "KOPIE"
0016              MVI C,00    ; pomocná data
0018              JMP PRESUN

0020  PRESKOC 1:  LDA 1F02    ; celý blok naprogramován?
0023              CMP L
0024              JNZ TU      ; není
0027              JMP KONEC   ; ano

0030  KOMPARACE:  CALL VSTUP
0033              ORI 0C      ; vytvoření počáteční adresy EPROM
0035              MOV H,A,
0036              LXI D,1C00  ; poč.adresa RAM
0039              XCHG
003A              DCX H
003B              DCX D
003C  OPET:       INX H
003D              INX D
003E              DCX B       ; test délky bloku dat
003F              MOV A,C
0040              CMP B
0041              JZ ANO      ; data zapsána správně
0044              LDAX D
0045              CMP M       ; není chyba?
0046              JNZ OPET    ; testuj další bajt
0049              MOV A,L     ; číslo chybného bajt
004A              STA 1FFA
004D              CALL 00FB   ; zobraz číslo chybného bajtu

0055  VSTUP:      LDA 1F02
0058              MOV C,A     ; nastavení čítače cyklu
0059              MVI B,00
005B              INX B
005C              INX B
005D              LHLD 1F00
0060              MOV A,H     ; utvoř vyšší bajt počáteční
                                adresy, přepiš z pozice 1000
                                do RAM

0070  PREPIS 1:   CALL VSTUP
0073              ORI 10      ; realizace adresy "ORIGINAL"
0075              JMP VYSTUP

0080  PREPIS 2:   CALL VSTUP  ; realizace adresy
                                program!EPROM
0083              ORI 0C,
0085              JMP VYSTUP

0090  VYSTUP:     MOV H,A
0091              LXI D,1C00
0094              DCX H
0095              DCX D
0096  TAM:        INX H
0097              INX D
0098              DCX B
0099              MOV A,C
009A              CMP B
009B              JZ ZOBRAZ   ; celý blok dat je přepsán?
009E              MOV A,M     ; ne
009F              XCHG
00A0              MOV M,A
00A1              XCHG
00A2              JMP TAM     ; pokračuj v přepisu dat

00AA  ANO:        LXI H,190C  ; zobraz "C dobrE"
00AD              SHLD 1FFF   ; ukládej kódy znaku do výst.
                                registrů PMI 80
00B0              LXI H,110D
00B3              SHLD 1FF1
00B6              LXI H,120B
00B9              SHLD 1FF3
00BC              LXI H,190E
00BF              SHLD 1FF5
00C2              MVI A,19
00C4              STA 1FF7
00C7              CALL 0116   ; podprogram "OUTKE" PMI 80

00D0  ZOBRAZ:     LXI H,1919  ; zobraz "END"
00D3              SHLD 1FFF   ; ukládej kódy znaku do výst.
                                registru PMI 80
00D6              LXI H,0E19
00D9              SHLD 1FF1
00DC              LXI H,0D1B
00DF              SHLD 1FF3
00E2              LXI H,1919
00E5              SHLD 1FF5
00E8              MVI A,19
00EA              STA 1FF7
00ED              CALL 0116   ; podprogram "OUTKE" PMI 80

00F0  PRESUN:     MVI A,88    ; programování 82 55
00F2              OUT FB
00F4              MVI A,80    ; PB7 na úroveň "H"
00F6              OUT F9
00F8              XCHG
00F9              DCX H
00FA              DCX D
00FB  TU:         MVI B,80
00FD              INX H       ; přechod na další adresu
00FE              INX D
00FF              MOV A,M     ; přesun dat z RAM
0100              CPI FF
0102              JZ PRESKOC  ; neprogramuj "FF"
0105  SEM:        XCHG
0106              MOV M,A     ; cyklus programování
                                jednoho bajtu dat
0107              DCR B
0108              JNZ SEM     ; je cyklus ukončen?
010B              XCHG,ANO
010C  PRESKOC:    MOV A,C     ; je požadován program
                                nebo kopie?
010D              CPI FF
010F              JZ PRESKOC 1; program
0112              NOP
0113              NOP
0114              MOV A,L     ; kopie
0115              CPI FF      ; test ukončení přepisu
0117              JNZ TU
011A              MOV A,H
011B              CPI 13
011D              JNZ TU      ; celá paměť překopírovaná?
0120  KONEC:      MVI A,00    ; ano
0122              OUT F9
0124              JMP ZOBRAZ  ; zobraz nápis "END"
```

Nyní si povšimneme složitějšího případu, když přechází blok dat svojí délkou přes dvě oblasti. V tom případě musíme program rozdělit na dva dílčí úseky. Např. blok dat z prvního příkladu chceme umístit do EPROM počínaje adresou 00FB. Podle **tabulky 1** je zřejmé, že se přechází z oblasti 1 do oblasti 2. Provedeme posloupnost následujících kroků:

1) Určíme délku bloku dat, který lze ještě umístit do zóny 1 pomocí vztahu:
FF – nižší bajt počáteční ukládací adresy + 01.
Tedy FF – FB + 01 = 05.

2) Buňky v RAM obsadíme následovně:
1C00 F3
1C01 3E
1C02 20
1C03 06
1C04 80

3) V souladu s tab. 1 zapíšeme na pomocné buňky:
1F00 FB (nižší bajt počáteční ukládací adresy)
1F01 20
1F02 04 (délka bloku dat)

4) Odstartujeme program PROGRAM.

5) Zbylá data zapíšeme do RAM opět počínaje buňkou 1C00:
1C00 21
1C01 FF
1C02 1E
1C03 23
1C04 00
1C05 76

6) Počáteční ukládací adresa je nyní 0100 a v souladu s **tab. 1** uložíme na:
1F00 00
1F01 21
1F02 05

7) Odstartujeme program PROGRAM příkazem G.

Výpis 2. Obsah pomocné paměti EPROM s řídicím programem (914–V2)

```
0000  0E FF 2A 00 1F 11 00 1C C3 F0 04 FF FF FF FF FF
0010  11 00 10 21 00 20 0E 00 C3 F0 04 FF FF FF FF FF
0020  3A 02 1F BD C2 FB 04 C3 20 05 FF FF FF FF FF FF
0030  CD 55 04 F6 0C 67 11 00 1C EB 2B 1B 23 13 0B 79
0040  B8 CA AA 04 1A BE CA 3C 04 7D 32 FA 1F CD FB 00
0050  FF FF FF FF FF 3A 02 1F 4F 06 00 03 03 2A 00 1F
0060  7C C9 FF FF FF FF FF FF FF FF FF FF FF FF FF FF
0070  CD 55 04 F6 10 C3 90 04 FF FF FF FF FF FF FF FF
0080  CD 55 04 F6 0C C3 90 04 FF FF FF FF FF FF FF FF
0090  67 11 00 1C 2B 1B 23 13 0B 79 B8 CA D0 04 7E EB
00A0  77 EB C3 96 04 FF FF FF FF FF 21 0C 19 22 EF 1F
00B0  21 0D 11 22 F1 1F 21 0B 12 22 F3 1F 21 0E 19 22
00C0  F5 1F 3E 19 32 F7 1F CD 16 01 FF FF FF FF FF FF
00D0  21 19 19 22 EF 1F 21 19 0E 22 F1 1F 21 1B 0D 22
00E0  F3 1F 21 19 19 22 F5 1F 3E 19 32 F7 1F CD 16 01
00F0  3E 88 D3 FB 3E 80 D3 F9 EB 2B 1B 06 80 23 13 7E
0100  FE FF CA 0C 05 EB 77 05 C2 06 05 EB 79 FE FF CA
0110  20 04 00 00 7D FE FF C2 FB 04 7C FE 13 C2 FB 04
0120  3E 00 D3 F9 C3 D0 04 FF FF FF FF FF FF FF FF FF
```

Seznam součástek

R1,R7	TP 011	22 kΩ
R2,R4	TR 151	12 kΩ
R3,R5	TR 151	390 Ω
R6	TR 151	5,6 kΩ
R8	TR 151	6,8 kΩ
R9–R12	TR 151	4,7 kΩ
C1,C3	TK 782	150 nF
C2	TK 782	10 nF
T1, T2, T4, T6, T3, T7, T8		KC507 KC508
T5		KSY63

Při přepisu z paměti EPROM v pozici ORIGINAL do paměti RAM je postup úplně shodný s tím, že na 1F00 ukládáme opět nižší bajt počáteční adresy, tentokrát však přepisovaného bloku dat, na 1F01 vyšší bajt této adresy zvětšený opět o + 20H a na 1F02 opět délku přepisovaného bloku dat.

Literatura

[1] Programátor paměti EPROM řízený mikropočítačem. ST8/81.

OSCILOSKOP ZO ZX SPECTRA

Jindřich Vídenský, *Lamač-Podháj 55, 841 03 Bratislava*

Program bol vytvorený pre využitie AD prevodníka s mikropočítačom v elektrotechnickej praxi ako jednoduchého nf pamäťového osciloskopu. Umožňuje merať v ôsmich pamäťových rozsahoch signály do 15 kHz.

Po nahraní programu do počítača sa zobrazí základné menu:

F . . . Zmena farieb INK, PAPER, BORDER.
Z . . . Zmena riadiaceho slova, jeho adresy a adresy vstupného portu.
P . . . Zmena adresy vlastného podprogramu, ľubovoľnej rutiny, ktorá sa dá vyvolať počas behu osciloskopu. Môže to byť napr. podprogram pre Hardcopy.
C . . . Zmena synchronizačnej hodnoty (viď ďalej).
S . . . Štart osciloskopu. Stlačením tejto klávesy sa vypíše

hlavné menu s týmito možnosťami

1–8 Stlačením jednej z týchto kláves vyberieme časový rozsah a zároveň sa spustí zobrazovací režim osciloskopu, v ktorom prebieha vzorkovanie signálu a jeho zobrazovanie.
Z . . . V zobrazovacom režime sa vykreslí na obrazovku sieťka 5×10.
X . . . Sieťka sa nebude vykreslovať.
C . . . Program po vykreslení priebehu čaká na stlačenie ľubovolnej klávesy, čím sa obraz na obrazovke zastaví a je možné napr. presne odčítať amplitúdu, alebo urobiť kópiu obrazovky.
V . . . Program nečaká na stlačenie klávesy, takže obraz sa mení plynule, podobne ako na obyčajnom osciloskope.
B . . . Vzorkovanie bude prebiehať bez softwarovej synchronizacie, viď N.
N . . . Synchronizacia priebehu vzorkovania prechodom signálu cez hodnotu nastavenú v základnom menu klávesou C. Program čaká, kým signál nedosiahne nastavenú hodnotu, zisti či priebeh rastie a ak áno, potom ho zobrazí. Tým sa dosiahne, že signál nebehá po obrazovke, ale vždy začína v určitej hodnote. Využiť sa dá tento režim hlavne pri periodických signáloch.
M . . . Tak isto, ako pri N, ale synchronizácia klesajúcim priebehom.
W . . . Návrat do základného menu.
P . . . Stlačením tejto klávesy počas zobrazovacieho režimu sa vykoná odskok na vlastný podprogram, ukončený inštrukciou RET. Zmenu adresy v základnom menu je vhodné urobiť len vtedy, ak je podprogram skutočne v pamäti, ináč hrozí zrútenie systému pri náhodnom stlačení P. Podprogram je treba umiestiť nad adresu 41500.
Q . . . Po stlačení Q sa dá vrátiť do tohoto menu zo zobrazovacieho režimu.

Program pracuje na princípe navzorkovania signálu do pamäti tak, aby dĺžka vzorky zodpovedala danému časovému rozsahu. Preto každému rozsahu zodpovedá jeden z časovacích podprogramov F01 až F680. Tieto zaisťujú pravidelné intervaly medzi jednotlivými vzorkami signálu. Touto časťou je limitovaná aj najvyššia frekvencia, ktorú je možné zobraziť, rešp. ešte odčítať na obrazovke. Na obrázku je príklad signálu 15 kHz, ktorý je ešte rozlíšiteľný. Po navzorkovaní 256 hodnôt z AD prevodníka do pamäti sú vzorky prepočítané z intervalu 9 až 255 do intervalu 0 až 191, aby ich bolo možné zobraziť na obrazovku Spectra.

Program sa skladá z BASICu a strojového kódu. Neodporúčam robiť v ňom veľké zmeny, hlavne riadok 700 musí zostať presne tak ak je, tzn. vcelku, dodržať všetky čiarky a pred riadkom 700 sa nesmie vyskytnúť iný riadok s príkazom DATA.

Strojový kód je písaný v assembleri, je odladený systémom MRS. Odporúčam nerobiť v ňom žiadne zmeny, preložiť ho od adresy 40000, pretože je modifikovaný príkazmi POKE z BASICu. Takisto je treba v assembleri preložiť aj inštrukcie NOP, ktoré sa na prvý pohľad môžu zdať zbytočné. Pretože je výpis programu vytlačený malými písmenami, je treba dát pozor pri prepisovaní na zámenu písmena l s číslicou 1.

Technické predpoklady

Je treba použiť osembitový A/D prevodník, ktorý má automatický štart prevodu a vyrovnávaciu pamäť. Vhodný je napríklad prevodník zo zelenej prílohy AR Mikroelektronika 1988. Rychlosť prevodu by však mala byť menšia ako 5,7 μs. Je daná najrýchlejším rozsahom a najrýchlejšou inštrukciou blokového prenosu s opakovaním INIR. Pri použití prevodníka s menším počtom bitov ako 8 sa zbytočne zmenšuje rozlišovacia schopnosť osciloskopu a použitie napr. 12 bitového prevodniku s využitím jeho horných 8 bitov je síce možné, avšak je zbytočným luxusom vzhladom k rozlišovacím schopnostiam obrazovky Spectra. Citlivosť osciloskopu je daná citlivosťou prevodníku.

Najvhodnejšie sú prevodníky typu MDAC 08, program nie je určený pre prevodníky s multiplexovým BCD výstupom typu C520. Vhodne je, aby vstup prevodniku bol bipolárny, vzhladom k potrebe sledovať nf signály.

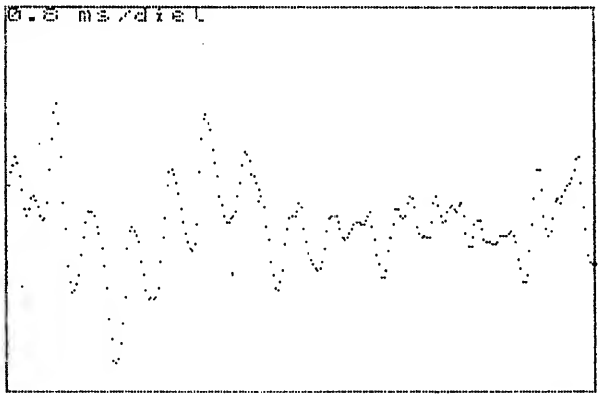
Na priložených kópiách obrazovky sú nasnímané rôzne priebehy v rôznych časových rozsahoch. Je z nich vidieť príklad použitia synchronizácie priebehu prechodom cez "nulovú" hodnotu, použitie sieťky, ako aj kvalita generátora „sinusového" priebehu.

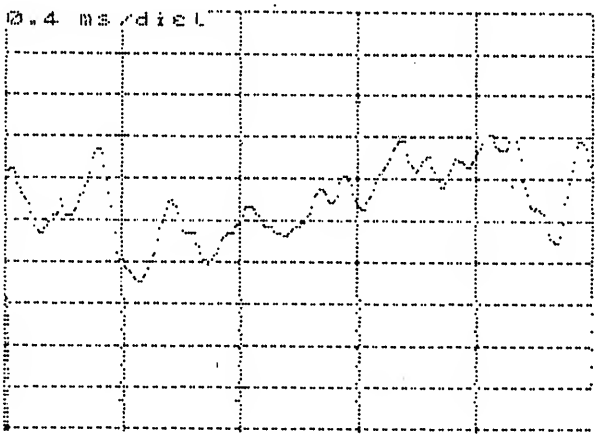
Záver

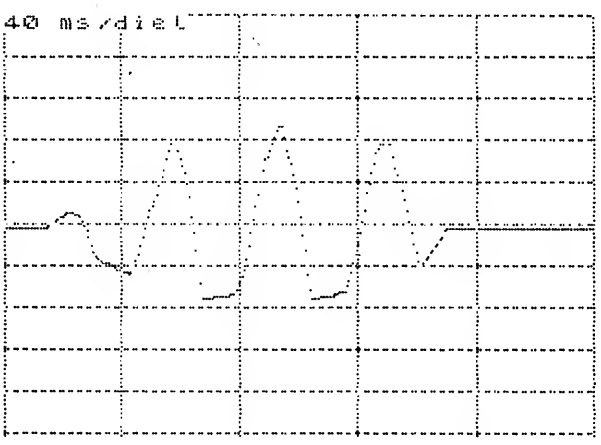
Program osciloskop nebol určený ako náhrada skutočného osciloskopu, ktorému sa svojimi parametrami nemôže rovnať, ale iba ako pomerne užitočná pomôcka pri nastavovaní jednoduchých elektrotechnických zariadení, na rôzne orientačné sledovanie analógových aj digitálnych signálov a ako náhrada za logickú sondu.

Výpis 1. Program v BASICu (928–V1)

```
 10 CLEAR 39999: BORDER 1: PAPE
R 1: INK 7: CLS : LOAD ""CODE 40
000: CLS : BORDER 0
 20 LET ash=40472: LET sh=127:
LET ash2=40478: LET ars=127: LET
 rs=130: LET vp=63: LET op=31
 40 GO TO 600
 50 PRINT AT 3,0;"1-ZMENA ADRES
Y RIADIACEHO SLOVA   PRE INTERFA
CE"
 60 PRINT AT 5,0;"2-ZMENA RIADI
ACEHO SLOVA PRE IN-  TERFACE"
 70 PRINT AT 8,9;"OUT(";ars;"),
";rs;"    "
 80 PRINT AT 10,0;"3-ZMENA ADRE
SY VSTUPNEHO PORTU    PREVODNIKU
 (A/D PREVOD)"
 90 PRINT AT 13,13;"IN(";PEEK 4
0707;")   "
 100 PRINT AT 15,0;"4-ZMENA ADRE
SY VYSTUPNEHO PORTU   PREVODNIKU
 C D/A PREVOD NIE     JE NUTNY J
"
 110 PRINT AT 18,13;"OUT(";op;")
  "
 120 PRINT AT 21,0; BRIGHT 1;"R-
NAVRAT K MENU"
 130 IF INKEY$="1" THEN  INPUT a
rs: GO TO 50
 140 IF INKEY$="2" THEN  INPUT r
s: GO TO 50
 150 IF INKEY$="3" THEN  INPUT v
p: POKE 40707,vp: GO TO 50
 160 IF INKEY$="4" THEN  INPUT o
p: GO TO 50
 170 IF INKEY$="R" OR INKEY$="r"
 THEN  GO TO 460
 180 GO TO 130
 400 INPUT "INK ";A: IF A<0 OR A
>7 THEN  GO TO 400
 410 INK A
 420 INPUT "PAPER ";A: IF A<0 OR
 A>7 THEN  GO TO 420
 430 PAPER A
 440 INPUT "BORDER ";A: IF A<0 O
R A>7 THEN  GO TO 440
 450 BORDER A
 460 CLS
 600 PRINT AT 6,1;"F-ZMENA FARIE
B""" Z-ZMENA HODNOT PRE INTERFA
CE""" P-ZMENA ADRESY VLASTNEHO"
""   PODPROGRAMU""" C-ZMENA SYN
CHRONIZACNEJ HODNOTY""" BRIGHT 1
;" S-START OSCILOSKOPU
  "
 610 IF INKEY$="F" OR INKEY$="f"
 THEN  GO TO 400
 620 IF INKEY$="z" OR INKEY$="Z"
 THEN  CLS : GO TO 50
 630 IF INKEY$="S" OR INKEY$="s"
 THEN  GO TO 700
 640 IF INKEY$="C" OR INKEY$="c"
 THEN  INPUT "nova hodnota ";sh:
 POKE ash,sh: POKE ash2,sh
 650 IF INKEY$="P" OR INKEY$="p"
 THEN  INPUT "adresa nesmie byt
z intervalu   40000-41500 ";ppp:
 POKE 40372,INT (ppp/256): POKE
40371,ppp-256*INT (ppp/256): GO
TO 610
 660 GO TO 610
 700 DATA "1-   2  s/diel,2- 200
 ms/diel,3-  40 ms/diel,4-   8 m
s/diel,5-   2 ms/diel,6- 0.8 ms/
diel,7- 0.4 ms/diel,8- 0.3 ms/di
el,Z-so sietkou,X-bez sietky,C-c
aka na stlacenie klavesy,  po vy
kresleni,V-necaka na stlacenie k
lavesy,B-vzorkovanie bez synchro
nizacie,N-synchronizacia prechod
om cez,  nastavenu hodnotu stupa
jucim,  priebehom,M-synchronizac
ia klesajucim,  priebehom,W-navr
at do hlavneho menu,P-vystup na
vlastny podprogram,Q-navrat zo z
obrazovacieho,  rezimu do tohoto
 menu,          ♦HEAVYSOFT 1988"
 710 OUT ars,rs: RANDOMIZE USR 4
0000
 720 GO TO 460
1000 SAVE "OSCILOSKOP" LINE 10
```







Obr. 1. Ukázky vytisknutých průběhů (928–1)

Výpis 2. Rutina ve strojovém kódu (928–V2)

```
3       60bb            *a
?4       60bb                   org  40000
5       9c40
6       9c40            ;vstupna rutina,tlac menu
7       9c40            ;modifikacia programu
8       9c40 f3         osc    di
9       9c41 cd8a9d            call clear
10      9c44 cda09d            call vstup
11      9c47 cd419d            call data
12      9c4a e5                push hl
13      9c4b dde1              pop  ix
```

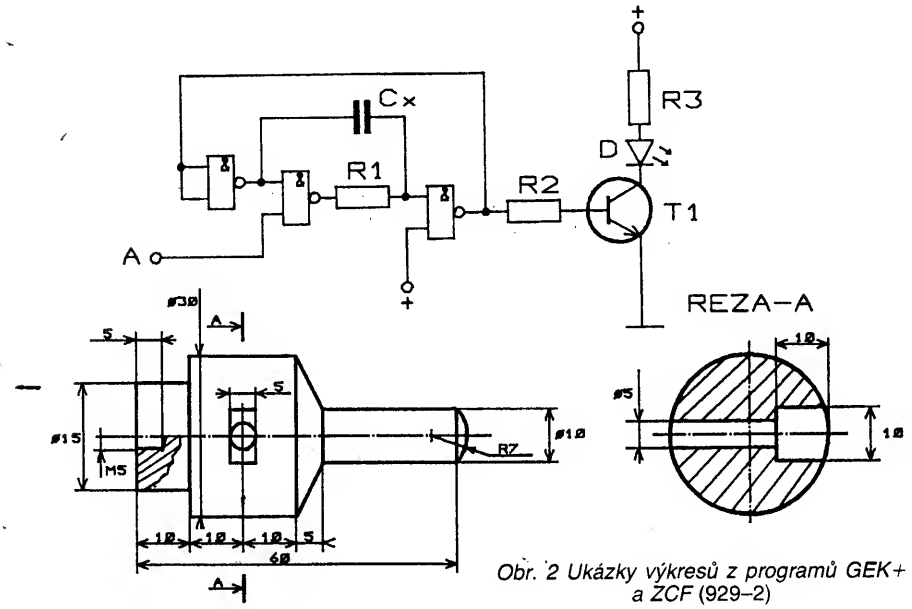
```
9c4d cd559d      scan1  call prnt
9c50 cd059e      scan   call keys
9c53 cd1b2d             call #2d1b
9c56 304b               jr   nc,vzor
9c58 fe5a               cp   'Z'
9c5a 2004               jr   nz,xx
9c5c af                 xor  a
9c5d 32b69d             ld   (gridd),a
9c60 fe58        xx     cp   'X'
9c62 2005               jr   nz,cc
9c64 3ec9               ld   a,#c9
9c66 32b69d             ld   (gridd),a
9c69 fe43        cc     cp   'C'
9c6b 2004               jr   nz,vv
9c6d af                 xor  a
9c6e 32049e             ld   (keyp),a
9c71 fe56        vv     cp   'V'
9c73 2005               jr   nz,bb
9c75 3ec9               ld   a,#c9
9c77 32049e             ld   (keyp),a
9c7a fe42        bb     cp   'B'
9c7c 2005               jr   nz,nn
9c7e 3ec9               ld   a,#c9
9c80 32109e             ld   (sync),a
9c83 fe4e        nn     cp   'N'
9c85 2009               jr   nz,mm
9c87 3e38               ld   a,#38
9c89 321f9e             ld   (updo),a
9c8c af                 xor  a
9c8d 32109e             ld   (sync),a
9c90 fe4d        mm     cp   'M'
9c92 2009               jr   nz,ww
9c94 3e30               ld   a,#30
9c96 321f9e             ld   (updo),a
9c99 af                 xor  a
9c9a 32109e             ld   (sync),a
9c9d fe57        ww     cp   'W'
9c9f 20af               jr   nz,scan
9ca1 fb                 ei
9ca2 c9                 ret
9ca3
9ca3                    ;vyber casovacich podprogramov
9ca3                    ;vykreslenie sietky
9ca3                    ;synchronizacia vzorkovania
9ca3                    ;vzorkovanie
9ca3 fe30        vzor   cp   '0'
9ca5 28a9              jr   z,scan
9ca7 fe39              cp   '9'
9ca9 28a5              jr   z,scan
9cab f5                push af
9cac cd8a9d            call clear
9caf f1                pop  af
9cb0 d631              sub  #31
9cb2 87                add  a,a
9cb3 212d9e            ld   hl,ftab
9cb6 1600              ld   d,0
9cb8 5f                ld   e,a
9cb9 d5                push de
9cba 19                add  hl,de
9cbb 5e                ld   e,(hl)
9cbc 23                inc  hl
9cbd 56                ld   d,(hl)
9cbe ed53e29c          ld   (cas+1),de
9cc2 d1                pop  de
9cc3 213d9e            ld   hl,ztab
9cc6 19                add  hl,de
9cc7 5e                ld   e,(hl)
9cc8 23                inc  hl
9cc9 56                ld   d,(hl)
9cca ed53d39c          ld   (text+2),de
9cce cdb69d            call gridd
9cd1 dd21999e    text  ld   ix,zf680
9cd5 cd559d            call prnt
9cd8 2108a0      samp  ld   hl,vzor1
9cdb 013f00      port3 ld   bc,#3f
9cde cd109e            call sync
9ce1 cda59e      cas   call f680
9ce4 cd2f9d            call zobr
9ce7 cd049e            call keyp
9cea cd229e            call qpress
9ced 38e9              jr   c,samp
9cef cdad9d            call cop
9cf2 cd8a9d            call clear
9cf5 3e18              ld   a,#18
9cf7 32069d            ld   (kres),a
9cfa 3e1f              ld   a,#1f
9cfc 32079d            ld   (kres+1),a
9cff dd2a2b9e          ld   ix,(dtad)
9d03 c34d9c            jp   scan1
9d06
9d06                    ;vykreslenie priebehu
9d06 181f        kres  jr   prvy
9d08 21049f            ld   hl,vzor2
9d0b 1600              ld   d,0
9d0d 7e          dls   ld   a,(hl)
9d0e e5                push hl
9d0f d5                push de
9d10 cdf19d            call prep
9d13 d1                pop  de
9d14 4a                ld   c,d
9d15 cdb022            call #22b0
9d18 3c                inc  a
9d19 47                ld   b,a
9d1a 3e01              ld   a,1
9d1c 0f          lop   rrca
9d1d 10fd              djnz lop
9d1f ae                xor  (hl)
9d20 77                ld   (hl),a
9d21 e1                pop  hl
9d22 14                inc  d
9d23 c8                ret  z
9d24 23                inc  hl
9d25 18e6              jr   dls
9d27 af          prvy  xor  a
9d28 32069d            ld   (kres),a
9d2b 32079d            ld   (kres+1),a
9d2e c9                ret
9d2f
9d2f cd069d      zobr  call kres
9d32 11049f            ld   de,vzor2
9d35 2108a0            ld   hl,vzor1
9d38 010001            ld   bc,256
9d3b edb0              ldir
9d3d cd069d            call kres
9d40 c9                ret
9d41
9d41 21d05c      data  ld   hl,23760
9d44 3ee4        dtr1  ld   a,228
9d46 23          dtr   inc  hl
9d47 be                cp   (hl)
9d48 20fc              jr   nz,dtr
9d4a 23                inc  hl
9d4b 3e22              ld   a,'"'
9d4d be                cp   (hl)
9d4e 20f4              jr   nz,dtr1
9d50 23                inc  hl
9d51 222b9e            ld   (dtad),hl
9d54 c9                ret
9d55
9d55                    ;rutina vypisu na screen
9d55 0600        prnt  ld   b,0
9d57 0e00        prnt2 ld   c,0
9d59 dd7e00      prnt1 ld   a,(ix+0)
9d5c fe22              cp   '"'
9d5e c8                ret  z
9d5f fe2c              cp   ','
9d61 2005              jr   nz,print
9d63 04                inc  b
9d64 dd23              inc  ix
9d66 18ef              jr   prnt2
9d68 c5          print push bc
9d69 11003c            ld   de,#3c00
9d6c 87                add  a,a
9d6d 6f                ld   l,a
9d6e 63                ld   h,e
9d6f 29                add  hl,hl
9d70 29                add  hl,hl
9d71 19                add  hl,de
9d72 eb                ex   de,hl
9d73 78                ld   a,b
9d74 42                ld   b,d
9d75 cd9e0e            call #0e9e
9d78 50                ld   d,b
9d79 0600              ld   b,#00
9d7b 09                add  hl,bc
9d7c 0608              ld   b,#08
9d7e 1a          slc   ld   a,(de)
9d7f 77                ld   (hl),a
9d80 1c                inc  e
9d81 24                inc  h
9d82 10fa              djnz slc
9d84 c1                pop  bc
9d85 0c                inc  c
9d86 dd23              inc  ix
9d88 18cf              jr   prnt1
9d8a
9d8a 210040      clear ld   hl,#4000
9d8d 110140            ld   de,#4001
9d90 010018            ld   bc,#1800
9d93 75                ld   (hl),l
9d94 edb0              ldir
9d96 3a8d5c            ld   a,(#5c8d)
9d99 77                ld   (hl),a
9d9a 01ff02            ld   bc,#02ff
9d9d edb0              ldir
9d9f c9                ret
9da0
9da0 3a039f      vstup ld   a,(brana)
9da3 32169e            ld   (port1+1),a
9da6 321c9e            ld   (port2+1),a
9da9 32dc9c            ld   (port3+1),a
9dac c9                ret
```

```
206   9dad
207   9dad 3edf       cop    ld   a,#df
208   9daf dbfe             in   a,(254)
209   9db1 1f               rra
210   9db2 d2b59d           jp   nc,obsl
211   9db5 c9         obsl  ret
212   9db6
213   9db6            ;podprogram pre sietku
214   9db6 00         gridd nop
215   9db7 af               xor  a
216   9db8 0e00       pkr   ld   c,#00
217   9dba f5               push af
218   9dbb cdb022           call #22b0
219   9dbe 36cc             ld   (hl),#cc
220   9dc0 e5               push hl
221   9dc1 d1               pop  de
222   9dc2 13               inc  de
223   9dc3 011f00           ld   bc,#1f
224   9dc6 edb0             ldir
225   9dc8 f1               pop  af
226   9dc9 c613             add  a,#13
227   9dcb fed1             cp   #d1
228   9dcd 20e9             jr   nz,pkr
229   9dcf 0e00             ld   c,#00
230   9dd1 3ebf       tt    ld   a,#bf
231   9dd3 cb47       sm    bit  0,a
232   9dd5 2810             jr   z,idr
233   9dd7 f5               push af
234   9dd8 c5               push bc
235   9dd9 cdb022           call #22b0
236   9ddc 3c               inc  a
237   9ddd 47               ld   b,a
238   9dde 3e01             ld   a,#01
239   9de0 0f         lp1   rrca
240   9de1 10fd             djnz lp1
241   9de3 ae               xor  (hl)
242   9de4 77               ld   (hl),a
243   9de5 c1               pop  bc
244   9de6 f1               pop  af
245   9de7 3d         idr   dec  a
246   9de8 20e9             jr   nz,sm
247   9dea 3e33             ld   a,#33
248   9dec 89               adc  a,c
249   9ded d8               ret  c
250   9dee 4f               ld   c,a
251   9def 18e0             jr   tt
252   9df1
253   9df1            ;rutina nasobenia 3/4
254   9df1 2600       prep  ld   h,#00
255   9df3 6f               ld   l,a
256   9df4 54               ld   d,h
257   9df5 5d               ld   e,l
258   9df6 29               add  hl,hl
259   9df7 19               add  hl,de
260   9df8 cb3c             srl  h
261   9dfa cb1d             rr   l
262   9dfc cb3c             srl  h
263   9dfe cb1d             rr   l
264   9e00 3ebf             ld   a,#bf
265   9e02 95               sub  l
266   9e03 c9               ret
267   9e04
268   9e04            ;vstup z klavesnice
269   9e04 00         keyp  nop
270   9e05 cd8e02     keys  call #028e
271   9e08 20fb             jr   nz,keys
272   9e0a cd1e03           call #031e
273   9e0d 30f6             jr   nc,keys
274   9e0f c9               ret
275   9e10            ;synchronizacia
276   9e10 00         sync  nop
277   9e11 cd229e           call qpress
278   9e14 d0               ret  nc
279   9e15 db3f       port1 in   a,(63)
280   9e17 fe7f       sync1 cp   127
281   9e19 20f5             jr   nz,sync
282   9e1b db3f       port2 in   a,(63)
283   9e1d fe7f             cp   127
284   9e1f 38ef       updo  jr   c,sync
285   9e21 c9               ret
286   9e22
287   9e22 cdad9d     qpress call cop
288   9e25 3efb             ld   a,#fb
289   9e27 dbfe             in   a,(254)
290   9e29 1f               rra
291   9e2a c9               ret
292   9e2b            ;tabulky
293   9e2b 0000       dtad  db   0,0
294   9e2d f09edd9e   ftab  dw   f01,f1,f5
            d09e
295   9e33 c39eb69e         dw   f25,f100
296   9e37 ad9ea89e         dw   f250,f500,f680
            a59e
297   9e3d 4d9e569e ztab    dw   zf01,zf1,zf5,zf25
            629e6d9e
298   9e45 779e819e         dw   zf100,zf250,zf500
            8d9e
299   9e4b 999e             dw   zf680
300   9e4d 3220732f   zf01  db   '2 s/diel"'
            6469656c
            22
301   9e56 32303020   zf1   db   '200 ms/diel"'
            6d732f64
            69656c22
302   9e62 3430206d   zf5   db   '40 ms/diel"'
            732f6469
            656c22
303   9e6d 38206d73   zf25  db   '8 ms/diel"'
            2f646965
            6c22
304   9e77 32206d73   zf100 db   '2 ms/diel"'
            2f646965
            6c22
305   9e81 302e3820   zf250 db   '0.8 ms/diel"'
            6d732f64
            69656c22
306   9e8d 302e3420   zf500 db   '0.4 ms/diel"'
            6d732f64
            69656c22
307   9e99 302e3320   zf680 db   '0.3 ms/diel"'
            6d732f64
            69656c22
308   9ea5            ;casovacie rutiny
309   9ea5 edb2       f680  inir
310   9ea7 c9               ret
311   9ea8 eda2       f500  ini
312   9eaa 20fc             jr   nz,#fc
313   9eac c9               ret
314   9ead eda2       f250  ini
315   9eaf f5               push af
316   9eb0 c600             add  a,#00
317   9eb2 f1               pop  af
318   9eb3 20f8             jr   nz,#f8
319   9eb5 c9               ret
320   9eb6 eda2       f100  ini
321   9eb8 f5               push af
322   9eb9 c5               push bc
323   9eba 0605             ld   b,#05
324   9ebc 10fe             djnz #fe
325   9ebe c1               pop  bc
326   9ebf f1               pop  af
327   9ec0 20f4             jr   nz,#f4
328   9ec2 c9               ret
329   9ec3 eda2       f25   ini
330   9ec5 f5               push af
331   9ec6 c5               push bc
332   9ec7 0625             ld   b,#25
333   9ec9 10fe             djnz #fe
334   9ecb c1               pop  bc
335   9ecc f1               pop  af
336   9ecd 20f4             jr   nz,#f4
337   9ecf c9               ret
338   9ed0 eda2       f5    ini
339   9ed2 f5               push af
340   9ed3 c5               push bc
341   9ed4 06cd             ld   b,#cd
342   9ed6 10fe             djnz #fe
343   9ed8 c1               pop  bc
344   9ed9 f1               pop  af
345   9eda 20f4             jr   nz,#f4
346   9edc c9               ret
347   9edd eda2       f1    ini
348   9edf f5               push af
349   9ee0 c5               push bc
350   9ee1 0600             ld   b,#00
351   9ee3 10fe             djnz #fe
352   9ee5 10fe             djnz #fe
353   9ee7 10fe             djnz #fe
354   9ee9 10fe             djnz #fe
355   9eeb c1               pop  bc
356   9eec f1               pop  af
357   9eed 20ee             jr   nz,#ee
358   9eef c9               ret
359   9ef0 eda2       f01   ini
360   9ef2 f5               push af
361   9ef3 c5               push bc
362   9ef4 0629             ld   b,#29
363   9ef6 c5               push bc
364   9ef7 0600             ld   b,#00
365   9ef9 10fe             djnz #fe
366   9efb c1               pop  bc
367   9efc 10f8             djnz #f8
368   9efe c1               pop  bc
369   9eff f1               pop  af
370   9f00 20ee             jr   nz,#ee
371   9f02 c9               ret
372   9f03 fe         brana db   254
373   9f04            ;buffer pre vzorky
374   9f04            vzor2 ds   260
375   a008            vzor1 ds   260
376   a10c
377   a10c                  end
```

PROGRAM PRO KRESLENÍ GEK+

Jan Věříš, *Leninova 268, 533 41 Lázně Bohdaneč*

V poslední době se začal mezi vlastníky domácích počítačů rozšiřovat zapisovač jako periferní zařízení. Rozlišovací schopnost běžných zapisovačů je podstatně větší než rozlišení na obrazovce počítače. Většina kreslicích programů pro ZX Spectrum však provádí kresbu přímo na obrazovku a nevytváří přitom žádný záznam o vzniku kresby. V tomto případě pak slouží zapisovač pouze jako pomalejší tiskárna a jeho hlavní přednosti nejsou využity. Aby bylo možno plně využít velkou rozlišovací schopnost zapisovače, je nutné, aby program zaznamenával kreslené objekty v tzv. vektorové formě do paměti (tj. zaznamenával pouze souřadnice koncových bodů a některé další důležité parametry s co největší přesností, podle kterých lze kresbu zpětné zrekonstruovat. Výsledný vektorový soubor lze pak vykreslit na souřadnicovém zapisovači s maximální možnou přesností.



Obr. 2 Ukázky výkresů z programů GEK+ a ZCF (929–2)

1. Základní filozofie programu GEK+

Kreslicí plocha programu GEK+ má rozměr 20000×20000 základních jednotek (1 základní jednotka = 1 pixel při zobrazení ZOOM 1:1 viz dále). Na této pracovní ploše lze umisťovat základní objekty – čáry (LINE), kruhové oblouky (ARC), a text o maximální délce 5 znaků (TEXT).

Jedna kreslicí plocha se nazývá blok. Lze otevřít (OPEN) až 60 nezávislých kreslicích ploch – bloků. Každý otevřený blok je v podstatě dalším základním objektem, který lze tedy stejně jako čáru, text, oblouk umístit kamkoliv do jiného bloku. Umísťování bloků do sebe je omezeno pouze tak, aby nevznikla rekurze – tedy nelze vložit blok do sebe samého.

Obrazovka počítače je při běhu programu GEK+ rozdělena na tři samostatné části. V pravé části je trvale zobrazena nabídka – menu přístupných příkazů. Ve spodní části obrazovky je tzv. dialogová řádka, na které program zobrazuje jednoduchou nápovědu podle okamžité situace. Na zbytku obrazovky je zobrazen výřez z kreslicí plochy právě aktivního bloku s ohledem na zvolené zvětšení (ZOOM) a posun (PAN).

2. Ovládání programu

Program se ovládá pomocí osmi tlačítek klávesnice. Čtyři pro pohyb kurzoru (Q nahoru, A dolů, O vlevo, P vpravo). Čtyři tlačítka Z, X, M, N slouží pro výběr variant.

Po nahrání programu se zobrazí kurzor v podobě křížku uprostřed obrazovky. Při pohybu se jeho rychlost po skocích postupně zvyšuje. Po dosažení okraje obrazovky se kurzor zastaví. V horní části obrazovky se průběžně tiskne x-ová a y-ová vzdálenost kurzoru od bodu o souřadnicích 0,0 v milimetrech (1 základní jednotka = 0,125 mm, což odpovídá nejmenšímu kroku zapisovače).

Dialogová řádka je pomyslně rozdělena na čtyři části; každá část odpovídá jednomu tlačítku pro výběr varianty. Zleva to tedy jsou Z, X, N a M. Po spuštění programu tato řádka vypadá takto:

ERASE MENU TEXT LINE

Tlačítkem Z tedy vyvoláme funkci *ERASE*, tlačítkem N funkci *TEXT* a tlačítkem M funkci *LINE*. Stiskneme-li klávesu X, dostaneme se do menu. V dialogové řádce se objeví tato zpráva:

SELECT CREATE DEFINE SELECT

Nyní můžeme posunovat barevným kurzorem v pravé části obrazovky po jednotlivých příkazech menu pomocí kláves Q, A – pomalý posun a O, P – posun po stránkách. Vybranou funkci potvrdíme klávesou Z nebo M. Pomocí klávesy X můžeme kdykoliv opustit toto menu bez provedení příkazu. Při častém užívání některých funkcí by stálé hledání v menu zbytečně zpomalovalo práci, proto je možné předdefinovat tři funkce na tlačítka Z, N a M. Tyto funkce jsou pak kdykoliv přístupné stiskem příslušného tlačítka bez nutnosti jejich výběru z menu. Postup je následující: nastavíme kurzor na vybranou funkci a stiskneme N *(=DEFINE)*, potom stiskneme jednu z kláves Z, N nebo M. Nyní již je vybraná funkce přístupná přes vybranou klávesu přímo bez volání menu. Jméno definované funkce se také zobrazí v příslušné kolonce dialogového řádku.

3. Operace s bloky

3.1 OPEN

Po spuštění programu není otevřen žádný blok – neexistuje tedy kreslicí plocha, do které bychom mohli umisťovat objekty. Pro otevření nové kreslicí plochy slouží funkce OPEN. Aktivujeme-li tuto funkci, program nejprve kontroluje, zda lze otevřít další blok (počet bloků je maximálně 60). V případě, že je otevřeno již 60 bloků, program ohlásí chybu „Too many blocks". Je-li vše v pořádku, program požaduje zadání šestiznakového jména otevíraného bloku. Po napsání jména stiskneme ENTER. V této chvíli je nový blok otevřen a jeho jméno se stalo součástí menu.

3.2 CURR

V jednom okamžiku lze samozřejmě kreslit pouze do jednoho tzv. aktivního bloku. Chceme-li kreslit či provádět změny v jiném již otevřeném bloku, vyvoláme funkci CURR. Nyní se v pravé části obrazovky zobrazí pouze poslední část menu se jmény otevřených bloků a v dialogovém řádku se vypíše nápověda:

Backgr. Current block

Vybereme blok, který se má stát aktivním blokem a potvrdíme volbu klávesou X, N nebo M. Stiskneme-li klávesu Z, vybraný blok se vykreslí méně výrazně jako pozadí pod následně vybraný aktivní blok. Toto je výhodné např. při kreslení dvoustranných desek plošných spojů apod.

Pozn.:

I. Blok otevřený funkcí OPEN se zároveň stává i aktivním blokem a taktéž základním objektem (viz kapitola 2).

II. Vybereme-li kurzorem prázdný řádek v nabídce, program se chová tak, jako bychom vybrali funkci popř. blok, jehož jméno je zobrazené první.

4. Kreslení objektů

V programu GEK+ existují čtyři druhy základních objektů. Jsou to čáry (příkaz *LINE*), kruhové oblouky *(ARC)*, text *(TEXT)*. Každý otevřený blok se jeví z hlediska druhých bloků také jako objekt – tj. lze ho umístit na libovolné místo jiného bloku a to i několikrát.

4.1 Čáry – příkaz LINE

Kurzor umístíme do počátečního bodu čáry a vyvoláme funkci LINE, v dialogovém řádku se objeví zpráva To point, kurzor tedy nastavíme na koncový bod čáry a potvrdíme (klávesy Z, X, N, M). Na obrazovce se nakreslí rovný úsek čáry specifikované tloušťky (viz kapitola 5.4), v dialogové řádce se zobrazí:

END To point

Klávesou Z tedy můžeme funkci *LINE* opustit nebo lze nastavit kurzor do dalšího bodu a potvrdit (X, N nebo M). Tím je možné kreslit další úseky lomené čáry.

4.2 Kruhové oblouky – příkaz ARC

Vyvoláme funkci *ARC*, v dialogové řádce se objeví: Center point, nastavíme tedy kurzor na střed kruhového oblouku a potvrdíme (Z, X, N nebo M), v dialogové řádce se objeví: Start point, nastavíme kurzor na počáteční bod oblouku a potvrdíme. Nyní program požaduje zadání úhlu opsaného obloukem ve stupních (kladný údaj – oblouk bude pokračovat od počátečního bodu proti směru hodinových ručiček, záporný údaj – po směru hodinových ručiček). Zadaný číselný údaj potvrdíme stiskem klávesy ENTER.

4.3 Text příkaz TEXT

Kurzor nastavíme do bodu, kde se má nacházet levý dolní roh řádky textu. Vyvoláme funkci *TEXT*. Program požaduje zadání textového řetězce o délce právě 5 znaků. Pokud je text kratší, je nutné jej doplnit mezerami ! ! Text potvrdíme ENTER. Nyní zadáme výšku textu v milimetrech a opět potvrdíme ENTER. Pro ušetření místa v paměti a pro zrychlení programu se na obrazovce nevypíše text, ale pouze rámeček, který respektuje délku a rozměry textu.

4.4 Vkládání bloků

Umístíme kurzor na místo, kde se má nacházet bod o souřadnicích 0,0 ve vkládaném bloku. Nyní v menu nastavíme kurzor na jméno bloku, jež chceme vložit do právě aktivního bloku a potvrdíme stejně jako jakoukoliv jinou funkci (Z nebo M).

Pozn.:

I. Jméno bloku lze samozřejmě stejně jako jinou funkci předdefinovat na klávesy Z, N nebo M.

II. Program nepřipouští rekurzivní vkládání bloků do sebe. Nelze tedy vložit blok 1 do bloku 1 – tomu program zabrání, ale je-li blok 1 vložen do bloku 2 a dojde k pokusu o vložení bloku 2 do bloku 1, program to nerozpozná a havaruje ! !

5. Opravy výkresu

I při nejpečlivější práci se můžeme zmýlit. Pro opravy výkresu slouží v programu GEK+ tři funkce: vymazávání *(ERASE)*, posun *(MOVE)*, a kopírování *(COPY)*.

5.1 Vymazávání – příkaz ERASE

Touto funkcí lze vymazat všechny objekty, které alespoň částečně zasahují do definované obdélníkové oblasti. Pokud je do právě aktivního bloku vložen jiný blok, je tento chápán jako jediný objekt, tedy – zasahuje-li do vymezené oblasti alespoň jeden objekt z vloženého bloku, bude celý tento vložený blok vymazán. Po vyvolání funkce ERASE označíme nejprve levý dolní roh oblasti, kterou chceme vymazat. Potom označíme pravý horní roh této oblasti. Po potvrzení obou bodů program jakoby „v duchu" musí nakreslit znova všechny objekty a vybrané odstranit z paměti. Zdánlivě se chvilku nic neděje. Když je program hotov, vymaže obrazovku a nakreslí znovu aktivní blok již bez vymazaných objektů. Pozor! V případě složité kresby může být tato činnost velmi časově náročná.

Pozn.: Při znovuvykreslení aktivního bloku funkcí ERASE se již nevykreslí případné pozadí (viz kapitola 3. 2).

5.2 Přesun MOVE

Tato funkce umožňuje přesunout výřez právě aktivního bloku na jiné místo tohoto bloku. Postup je obdobný jako u fukce ERASE. Po zadání přesouvané oblasti program chvíli „přemýšlí", poté jsme dotázáni na polohu bodu, do kterého se přesune původně označený levý dolní roh přesouvané oblasti. Další činnost programu je obdobná funkci *ERASE*.

Pozn.: viz 5.1

5.3 Kopírování COPY

Tato funkce okopíruje objekty z definované oblasti na jiné místo. Postup je obdobný jako u funkce *MOVE*.

5.4 Příkaz LINE TYPE (L. TYPE)

Umožňuje zadat počet paralelně jdoucích čar a jejich vzdálenost, které se vykreslí při použití *LINE* nebo *ARC*. Pomocí tohoto příkazu tedy lze kreslit různě tlusté čáry.

Nejprve zadáme počet paralelně jdoucích čar (musí být větší nebo roven jedné) a potom vzdálenost mezi jednotlivými čarami v základních jednotkách (1 základní jednotka = 1 pixel při zobrazení *ZOOM 1:1* viz níže).

Definovaný typ čáry platí až do dalšího předefinování. Po nahrání programu je automaticky nastaven typ: počet čar = 1, vzdálenost = 0 (pokud je počet čar roven jedné, nemá údaj vzdálenosti žádný význam).

6. Skupina příkazů pro řízení zobrazování

Protože kreslicí plocha programu GEK+ je v porovnání s obrazovkou počítače daleko větší, není vždy možné zobrazit na obrazovce celý výkres s dostatečnou přesností. Na obrazovce je vidět pouze výřez okolí definovaného bodu kreslicí plochy (po spuštění je to okolí bodu 0,0) v definovaném zmenšení. Pomocí funkce ZOOM je možné měnit zmenšení kresby, funkcí PAN lze definovat bod, jehož okolí chceme zobrazovat.

6.1 Příkaz ZOOM

Umožňuje měnit měřítko zobrazení na obrazovce. Souřadnice objektů jsou v paměti počítače uloženy s konečnou přesností. Zadáme-li měřítko 1:1, bude vzdálenost dvou sousedních pixelů na obrazovce zároveň nejmenší možnou vzdáleností, kterou může program GEK+ rozlišit. Tato vzdálenost také souhlasí s rozlišením zapisovače, které je 0,125 mm. Navývá se základní jednotka.

Po vyvolání příkazu ZOOM se v dialogové řádce zobrazí výzva: Scale 1: a program čeká na zadání čísla z intervalu 1 až 99 včetně, které říká, kolik základních jednotek připadne na jeden pixel. Jsou povolena i desetinná čísla. Po zadání hodnoty program znovuvykreslí výkres v zadaném měřítku.

6.2 Příkaz PAN

Na obrazovce je zobrazena část kreslicí plochy v okolí bodu 0,0. Chceme-li pracovat na jiné části výkresu, která je momentálně mimo rámec obrazovky, použijeme příkaz PAN. Po jeho vyvolání program vyžaduje zadání souřadnic bodu, který má být ve středu obrazovky. Po zadání souřadnic program smaže obrazovku a znovuvykreslí okolí definovaného bodu. Při kreslení se samozřejmě respektuje dříve zadané zmenšení.

7. Příkazy pro spolupráci s magnetofonem

7.1 SAVE

Pomocí tohoto příkazu je možno uschovat na magnetofonový pásek jeden či několik bloků.

Po vyvolání tohoto příkazu program nejprve požaduje zadání prvního nahrávaného bloku. V nabídce tedy vybereme kurzorem první blok, který chceme nahrát a potvrdíme klávesou Z, X, N nebo M. Stejným způsobem označíme i poslední nahrávaný blok. Nyní program požaduje zadání jména nahrávky. Tento název musí mít přesně 5 znaků! Po odeslání jména (ENTER) spustíme magnetofon. Nahrají se postupně tři soubory, které nesou všechny potřebné informace o označených blocích.

7.2 Příkaz LOAD

Pomocí tohoto příkazu lze nahrát do paměti počítače bloky dříve uchované příkazem SAVE. Po vyvolání příkazu LOAD zadáme pětiznakové jméno, které jsme použili při ukládání bloků na pásek pomocí SAVE. Poté spustíme magnetofon. Program automaticky řídí nahrání všech tří částí záznamu. Nahraný blok (bloky) se připojí za již otevřené bloky. Jestliže by při připojování bloků příkazem LOAD hrozilo překročení vymezené kapacity paměti, vypíše se hlášení *„No room for file"* a volba je zrušena.

8. Další důležité informace

8.1 Zatímco u klasických kreslicích programů je omezena pracovní plocha počtem zobrazitelných bodů na obrazovce, u programu GEK+ je kreslicí plocha prakticky neomezená (20000×20000 základních jednotek odpovídá přibližně rozměru výkresu 2,5×2,5 metrů při rozlišení 0,125 milimetrů), je však omezen celkový počet objektů, které je možno do kreslicí plochy (kreslicích ploch) umístit. Pro záznam objektů je vymezeno přibližně 8 kB operační paměti, což odpovídá přibližně 1500 úsekům rovných čar či 700 kruhovým obloukům. Pokud se právě zadávaný objekt již nevejde do paměti, program vypíše zprávu *„Memory full"*. Objekt se sice nakreslí na obrazovku, ale do paměti se již nezaznamená.

8.2 Vložíme-li blok jako objekt do jiného bloku, nezaznamenává se do paměti jméno tohoto bloku, ale pouze jeho pořadové číslo. To může ve spojení s příkazy SAVE a LOAD činit určité potíže.

Příklad:

Mějme nahraný na pásce soubor s informacemi o těchto třech blocích:

1. **CTVER** obsahuje čtverec vytvořený ze čtyř úseků rovné čáry,
2. **KOLO** obsahuje kruh vytvořený příkazem ARC,
3. **CTVK** obsahuje vnořené bloky CTVER. a KOLO.

Po nahrání programu GEK+ otevřeme příkazem *OPEN* první blok a nazveme jej KRIZEK, bude obsahovat např. křížek složený ze dvou rovných čar. Nyní použijeme příkaz *LOAD* a přihrajeme výše uvedený soubor bloků. Hlavní menu potom může vypadat například takto:

.
.
.

KRIZEK
CTVER
KOLO
CTVK

Podíváme-li se nyní příkazem *CURR* na bloky KRIZEK, CTVER. a KOLO, jsou v pořádku. Podíváme-li se však na blok CTVK, vidíme křížek a čtverec, ač zde byl původně čtverec a kruh. Blok CTVK obsahuje první a druhý blok – původně to byly bloky CTVER. a KOLO, nyní je však první blok KRIZEK a druhý CTVER. a tak se také zobrazí.

9. Popis vnitřní struktury programu

Zdrojový text programu GEK+ byl napsán v jazyce Pascal a přeložen kompilátorem hp113, což je upravená verze kompilátoru HP4T1.6 fy HiSoft doplněná o některé grafické procedury.

Program lze pro zjednodušení rozdělit na několik částí: blok procedur pro ovládání datového souboru s uloženými objekty, blok funkcí pro kreslení objektů na obrazovku, blok procedur, které provádějí uživatelem vybrané funkce, část, která řídí komunikaci s uživatelem (ovládání menu, kurzoru atd.) a nakonec část programu, která řídí součinnost všech těchto částí.

Funkce pro kreslení objektů:

LINETL vykreslí čáru dle zadaných parametrů a platného měřítka a posunutí. Výstupem funkce je log. hodnota typu Boolean, která říká, zda nakreslená čára zasahuje do definovatelného okénka (používá se při MOVE, COPY a ERASE).

ARCKRES kreslení kruhového oblouku podle zadaného středu, počátku, opsaného úhlu a typu čáry. Ostatní viz LINETL.

TEXTKRES kreslení rámečku namísto textu.

KRESLI kreslení všech objektů, které obsahuje parametry specifikovaný blok. Funkce rekurzivně volá sama sebe, je-li do kresleného bloku vložen další blok. Ostatní viz LINETL.

Procedury pro ovládání datového souboru

Protože použitý překladač neumožňuje vytvořit vyhovující strukturu dynamických proměnných pro ukládání údajů o kreslených objektech, bylo nutné vytvořit procedury, které udržují speciální datové pole v paměti pro tyto objekty. Toto pole je tvořeno jednak samostatným úsekem paměti o délce asi 8 kB a dále skupinou ukazatelů na začátky jednotlivých podúseků v datovém poli. Struktura datového pole je na **obr. 1.**

Všechny ukazatele jsou soustředěny v poli UK, kromě toho jsou v tomto poli též uloženy údaje o počtu objektů v jednotlivých oblastech. Práci s informacemi o objektech umožňují tyto procedury a funkce:

POCETPRVKU vrací počet objektů uložených v specifikované oblasti.

NPRVEK vydá n-tý prvek ze zadané oblasti.

VLOZ vloží prvek do zadané oblasti.

SMAZ smaže vybraný prvek v zadané oblasti.

ZMEN změní údaje u specifikovaného prvku v zadané oblasti.

Zadaná oblast – je to oblast, ve které jsou uloženy informace o objektech stejného typu. Tato oblast je definována číslem bloku a typem oblasti (LINE, ARC . . .), což jsou parametry každé procedury.

Komunikace s uživatelem

Funkce **CPRIK** tato funkce řídí pohyb kurzoru, výběr z menu a předefinování uživatelských kláves. Vrací číslo příkazu, který byl vybrán uživatelem.

Funkce **POHYBKUR** řídí pohyb kurzoru po obrazovce, přepočet souřadnic, zobrazování souřadnic, osového kříže apod. Vrací znak odpovídající stisknuté klávese (kromě Q, A, O a P).

Procedury provádějící příkazy vyvolané z menu uživatelem: **PROPEN, PRCURR, PRERASE, PRMOVE, PRCOPY, PRLINETYPE, PRZOOM, PRPAN, PRSAVE, PRLOAD, PRLINE, PRARC, PRTEXT, PRBLOK.**

Řízení běhu programu

Vlastní prováděcí program nejprve volá funkci CPRIK a podle čísla požadovaného příkazu vyvolá příslušnou proceduru (viz procedury provádějící příkazy . . .). Po ukončení vyvolané procedury se činnost opakuje.

Programy GEK+ a ZCF jsou napsány v jazyce Pascal. Program GEK+ byl přeložen upraveným kompilátorem hp113 na počítači Spectrum 80 kB. Na Spectru 48 kB nelze program vzhledem k délce zdrojového textu a výsledného kódu přeložit. Program ZCF lze s jistými omezeními přeložit i v paměti 48 kB. Použitý kompilátor hp113 má zabudovány některé nestandardní procedury, které by bylo nutné při použití jiného překladače doprogramovat:

AT(R,S:INTEGER) přesune pozici pro tisk na řádek R a sloupec S.

PLOT(X,Y,A:INTEGER) vytiskne bod o souřadnicích X a Y, parametr A specifikuje barvu bodu.

DRFT body se automaticky spojují úsečkami.

DOT ruší účinek DRFT.

TOUT(Jméno:ARRAY (1..8) OF CHAR; START,DÉLKA:INTEGER) nahraje na magnetofon specifikovaný úsek paměti.

TIN(Jméno:ARRAY (1..8) OF CHAR; START:INTEGER nahraje data z magnetofonu do paměti.

Také v příkazu *WRITE* jsou použity některé nestandardní řídicí kódy:

CHR(21),CHR(B) ekvivalent OVER B v BASICu.

CHR(17),CHR(B) ekvivalent PAPER B v BASICu.

Při přenosu programu na jiný typ počítače by bylo nutné upravit:

– kontrolu souřadnic v procedurách LINEKRES, KRIZW, POHYBKUR podle rozlišovací schopnosti použitého počítače,
– souřadnice použité při volání procedury AT podle počtu zobrazitelných řádků a sloupců,
– adresu začátku a konce volné paměti pro data v konstantách OFFSET a TOPMEM.

Program ZCF je vystavěn kolem ovladače SUPERPLOT, který je dodáván se zapisovačem XY 4131. V případě použití jiného zapisovače a tím i ovladače by bylo nutné přeprogramovat procedury, které zajišťují předání parametrů z Pascalu ovladači:

MA, VA, AC, OG, MF, CS, SC, ODVYS5. Podrobně jsou tyto příkazy popsány v příručce k programovému vybavení zapisovače XY 4131.

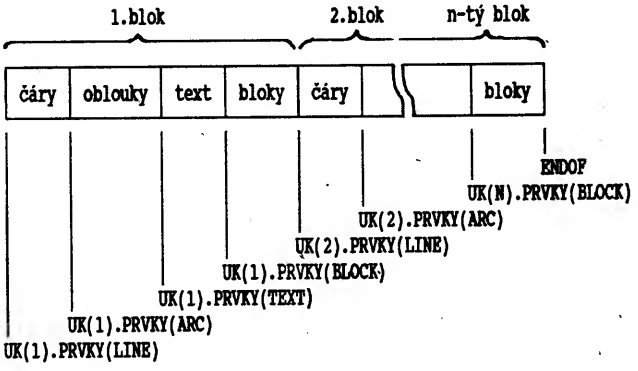
Pokud někdo nebude schopen program sám přeložit, může se obrátit na autora o nahrávku zkompilovaného programu.

10. Návaznost programu na kreslicí zařízení

Popis programu ZCF

Pro překreslení kresby zaznamenané programem GEK+ na souřadnicovém zapisovači byl napsán další kratší program. Je vystavěn kolem ovládacího programu SUPERPLOTT, který je dodáván společně se zapisovačem XY 4131. Bohužel jsem nikdy neměl možnost vyzkoušet tento program v originální sestavě (používám upravený SUPERPLOTT pro ovládání zapisovače pouze přes jediný osmibitový port), proto nemohu stoprocentně zaručit bezchybnou spolupráci se zapisovačem XY 4131.

Po nahráni programu příkazem LOAD"" se ZCF automaticky spustí. Nyní musíme do programu nahrát všechny bloky vytvořené



Obr. 1. Struktura datového pole (929–1)

programem GEK+ nutné pro správné zrekonstruování kresby (viz kapitola 8.2). Na dotaz „Name" napíšeme pětiznakové jméno souboru, který jsme před tím zaznamenali programem GEK+ příkazem SAVE a spustíme magnetofon. Postupně budou nahrány tři bloky dat s potřebnými údaji. Pokud je třeba přihrát ještě další soubor (soubory) jiného jména, stiskneme Y a postup se opakuje až máme všechny potřebné bloky nahrány v paměti.

Nyní program požaduje tyto údaje:

Scale měřítko zvětšení kresby (menší než 1 . . . zmenšení, větší než 1 . . . zvětšení).

Origin X,Y souřadnice vztažného bodu – na tento bod později nastavíme pisátko zapisovače a tím určíme polohu kresby na papíře.

Orientation 1 . . . standardní kresba,
2 . . . kresba bude zrcadlově otočena podle osy X
3 . . . kresba bude otočena o 90° podle počátku.

Po zadání všech těchto údajů proběhne inicializace zapisovače a pak program čeká, až pomocí kláves 5, 6, 7, 8 příp. +1 pro zrychlený pohyb nastavíme pisátko do bodu, jež má na papíře odpovídat výše zadanému vztažnému bodu. Jsme-li hotovi, stiskneme 0. V tomto okamžiku program zobrazí všechny nahrané bloky a jejich pořadová čísla a čeká na zadání počtu bloků, které chceme najednou vykreslit (ve většině případů jsou to jeden nebo dva např. oboustranné desky plošných spojů). Nyní postupně napíšeme pořadová čísla těchto bloků tak, jak jsou uvedena na obrazovce. Poté program vypíše zprávu „I am drawing" a kreslí specifikované bloky.

Pozn.:
I. Kreslení lze kdykoliv přerušit klávesou SPACE. Po stisku kterékoliv jiné klávesy kreslení pokračuje.
II. Volba Orientation pracuje správně pouze s upraveným ovládačem zapisovače. Je-li nahrán originální ovládač, volba je ignorována.

11. Ukázkový soubor PLSSP

Tento soubor nahrajeme do programu GEK+ (případně ZCF) příkazem LOAD (popř. výše uvedeným postupem) a zadáním jména PLSSP. Soubor obsahuje několik bloků:

MALYB	malý pájecí bod ∅ 1,25 mm,
VELKYB	velký pájecí bod ∅ 3 mm,
DIL14S	patice DIL 14 svisle,
DIL14V	patice DIL 14 vodorovně,
DIL16S	patice DIL 14 svisle,
DIL16V	patice DIL 14 vodorovně,
OZ	patice operačního zesilovače (kulatá),
XI, X2	pomocné bloky,
MRIZ5	pomocný rastr 5+5 mm,
STRSPO	příklad plošného spoje vytvořeného na
STRSOU	základě tohoto souboru pájecích bodů a patic.

Výpis 1. *Program GEK+* (929–V1)

```
PROGRAM GEK;
CONST OFFSET=#EA60;
      TOPMEM=#F450;
      PBLMAX=80;
      POCPRIK=23;
      JMCELKEM=83;
TYPE LINET=RECORD
           X,Y:INTEGER;END;
     ARCT=RECORD XC,YC,XS,YS,A,U:INTEGE
R;END;
     TEXTT=RECORD VELIKOST:CHAR;XP,YP:I
NTEGER;T:ARRAY[1..5] OF CHAR;END;
      OBJT=(LINE,ARC,TEXT,BLOK);
      UKAZATELE=ARRAY[1..PBLMAX] OF RE
CORD
                                      P
RVKY:ARRAY[LINE..BLOK] OF INTEGER;
                                      P
OCET:ARRAY[LINE..BLOK] OF INTEGER;
                                      E
ND;
     BLOKT=RECORD X,Y:INTEGER;CBL:CHAR;
END;
    NAMET=ARRAY[1..10] OF CHAR;
VAR X3,Y3,X4,Y4,PANX,PANY,GBL:INTEGER;
    SCALE:REAL;
    KRES:BOOLEAN;
    PLINE:LINET;PARC:ARCT;PTEXT:TEXTT;P
BLOK:BLOKT;
    UK:UKAZATELE;ENDOF:INTEGER;
     XCUR,YCUR,XCURS,YCURS,PBLAKT,MNUA,
MNUR:INTEGER;
     RYCHLOST:ARRAY[1..3] OF RECORD
                          XRY,YRY,P:INTE
GER;END;
    JMENA:ARRAY[1..JMCELKEM,1..6] OF CH
AR;
    DP:ARRAY[1..3] OF INTEGER;
    CURBLOK,CURLT:INTEGER;
    CISLO:INTEGER;
FUNCTION LINEKRES(X1,Y1,X2,Y2:INTEGER):
BOOLEAN;
VAR M1,M2,XR,YR:REAL;
    N,M:INTEGER;
    B2:BOOLEAN;
BEGIN
 XR:=((X1+PANX)*SCALE);YR:=((Y1+PANY)*S
CALE);
 M1:=((X2+PANX)*SCALE);M2:=((Y2+PANY)*S
CALE);
 LINEKRES:=FALSE;
B2:=((XR<0) AND (M1<0)) OR ((XR>206) AN
D (M1>206));
B2:=B2 OR ((YR<0) AND (M2<0)) OR ((YR>1
75) AND (M2>175));
IF NOT B2 THEN BEGIN
M:=ROUND(SQRT(SQR(X2-X1)+SQR(Y2-Y1))*SC
ALE);
IF M=0 THEN M:=1;
 M1:=(X2-X1)/M*SCALE;M2:=(Y2-Y1)/M*SCAL
E;
 FOR N:=0 TO M DO BEGIN
  X1:=ROUND(XR);Y1:=ROUND(YR);
 IF (X1>=0) AND (X1<=206) AND (Y1>=0) A
ND (Y1<=175) AND KRES THEN PLOT(X1,Y1,0
);
  IF (X1>=X3) AND (X1<=X4) AND (Y1>=Y3)
 AND (Y1<=Y4) THEN LINEKRES:=TRUE;
  XR:=(XR+M1);YR:=(YR+M2);
 END;
 END;
END;
FUNCTION LINETL(X1,Y1,X2,Y2,A:INTEGER):
BOOLEAN;
VAR N,M:INTEGER;M1,M2,XR,YR:REAL;
BEGIN
 LINETL:=FALSE;
 N:=A MOD 100;A:=A DIV 100;
 M1:=SQRT(SQR(X2-X1)+SQR(Y2-Y1));
IF M1=0 THEN M1:=1;
 M2:=(Y2-Y1)/M1*A*(N-1)/2;
 M1:=(X2-X1)/M1*A*(N-1)/2;
 X2:=ROUND(X2-X1);XR:=(X1-M2);
 Y2:=ROUND(Y2-Y1);YR:=Y1+M1;
IF N>1 THEN BEGIN M1:=M1/((N-1)/2);M2:=
M2/((N-1)/2);END;
 FOR M:=1 TO N DO BEGIN
X1:=ROUND(XR);Y1:=ROUND(YR);
  IF LINEKRES(X1,Y1,X1+X2,Y1+Y2) THEN L
INETL:=TRUE;
  XR:=XR+M2;YR:=YR-M1;
 END;
END;
FUNCTION ARCKRES(AC:ARCT):BOOLEAN;
VAR C,PR,PA,DA:REAL;N,M,D,E:INTEGER;
BEGIN
 WITH AC DO BEGIN
  C:=U/1000;
  PR:=SQRT(SQR(XS-XC)+SQR(YS-YC));
  N:=ROUND(ABS(PR*SCALE*2*(C/6.28)));
  PA:=ARCTAN((YS-YC)/(XS-XC+0.01));
  IF (XS-XC)<0 THEN PA:=PA+3.1415;
  ARCKRES:=FALSE;
  IF N=0 THEN ARCKRES:=LINETL(XS,YS,XS,
YS,A)
       ELSE BEGIN
   DA:=C/N;
   FOR M:=1 TO N DO BEGIN
    D:=ROUND(PR*COS(PA+DA)+XC);E:=ROUND
(PR*SIN(PA+DA)+YC);
IF LINETL(ROUND(XC+PR*COS(PA)),ROUND(YC
+PR*SIN(PA)),D,E,A) THEN ARCKRES:=TRUE;
    PA:=PA+DA;
    END;
   END;
  END;
END;
FUNCTION TEXTKRES(TXT:TEXTT):BOOLEAN;
VAR SKV,SKD,M:INTEGER;
BEGIN
 WITH TXT DO BEGIN
  SKV:=ORD(VELIKOST)*4;SKD:=0;
  FOR M:=1 TO 5 DO BEGIN
   IF (T[M]<>' ') AND (T[M]<>CHR(13)) T
HEN SKD:=SKD+SKV;END;
  TEXTKRES:=FALSE;
  IF LINEKRES(XP,YP,XP+SKD,YP) THEN TEX
TKRES:=TRUE;
  IF LINEKRES(XP+SKD,YP,XP+SKD,YP+SKV)
THEN TEXTKRES:=TRUE;
  IF LINEKRES(XP+SKD,YP+SKV,XP,YP+SKV)
THEN TEXTKRES:=TRUE;
  IF LINEKRES(XP,YP+SKV,XP,YP) THEN TEX
TKRES:=TRUE;
 END;
END;
PROCEDURE LDIR(HL,DE,BC:INTEGER);
BEGIN
IF BC>0 THEN BEGIN
POKE (23600,HL);POKE (23602,DE);POKE (2
3604,BC);
INLINE (#2A,#30,#5C,#ED,#5B,#32,#5C,#ED
,#4B,#34,#5C);
INLINE (#ED,#B0);
END;
END;
PROCEDURE LDDR(HL,DE,BC:INTEGER);
BEGIN
IF BC>0 THEN BEGIN
POKE(23600,HL);POKE(23602,DE);POKE(2360
4,BC);
INLINE (#2A,#30,#5C,#ED,#5B,#32,#5C,#ED
,#4B,#34,#5C);
INLINE (#ED,#B8);
END;
END;
FUNCTION POCETPRVKU(BLOK:INTEGER;OBJ:OB
JT):INTEGER;
BEGIN
 POCETPRVKU:=UK[BLOK].POCET[OBJ];
END;
PROCEDURE POINTERS(BL,M:INTEGER;OBJ:OBJ
T);
VAR N:INTEGER;
    POBJ:OBJT;
BEGIN
IF OBJ<BLOK THEN BEGIN
 FOR POBJ:=SUCC(OBJ) TO BLOK DO
  UK[BL].PRVKY[POBJ]:=UK[BL].PRVKY[POBJ
]+M;
END;
IF BL<PBLMAX THEN BEGIN
 FOR N:=BL+1 TO PBLMAX DO BEGIN
  FOR POBJ:=LINE TO BLOK DO
   UK[N].PRVKY[POBJ]:=UK[N].PRVKY[POBJ]
+M;
 END;
END;
ENDOF:=ENDOF+M;
END;
PROCEDURE ADLEN(OBJ:OBJT;VAR N,M:INTEGE
R);
BEGIN
CASE OBJ OF
 LINE:BEGIN N:=ADDR(PLINE);M:=4;END;
 ARC:BEGIN N:=ADDR(PARC);M:=12;END;
```

```
 TEXT:BEGIN N:=ADDR(PTEXT);M:=10;END;
 BLOK:BEGIN N:=ADDR(PBLOK);M:=5;END
END;
END;
PROCEDURE NPRVEK(BL,O:INTEGER;OBJ:OBJT)
;
VAR N,M:INTEGER;
BEGIN
ADLEN(OBJ,N,M);
 LDIR((O-1)*M+UK[BL].PRVKY[OBJ],N,M);
END;
PROCEDURE VLOZ(BL:INTEGER;OBJ:OBJT);
VAR N,M:INTEGER;
BEGIN
IF ENDOF<TOPMEM THEN BEGIN
 ADLEN(OBJ,N,M);
 WITH UK[BL] DO BEGIN
  LDDR(ENDOF,ENDOF+M,ENDOF-PRVKY[OBJ]-M
*POCET[OBJ]+1);
  LDIR(N,PRVKY[OBJ]+M*POCET[OBJ],M);
  POCET[OBJ]:=POCET[OBJ]+1;
 END;
 POINTERS(BL,M,OBJ);END
ELSE BEGIN
AT(22,0);WRITE('Memory full !!!
');END;
END;
PROCEDURE SMAZ(BL,M:INTEGER;OBJ:OBJT);
VAR N,O:INTEGER;
BEGIN
ADLEN(OBJ,O,N);
WITH UK[BL] DO
LDIR(PRVKY[OBJ]+M*N,PRVKY[OBJ]+(M-1)*N,
ENDOF-PRVKY[OBJ]-M*N);
N:=-N;POINTERS(BL,N,OBJ);
UK[BL].POCET[OBJ]:=UK[BL].POCET[OBJ]-1;
END;
PROCEDURE ZMEN(BL,O:INTEGER;OBJ:OBJT);
VAR M,N:INTEGER;
BEGIN
 ADLEN(OBJ,N,M);
 LDIR(N,UK[BL].PRVKY[OBJ]+M*(O-1),M);
END;
FUNCTION BLOKKRES(BLK:BLOKT):BOOLEAN;
VAR X1,Y1,X2,Y2,M,N,A,BL:INTEGER;
BEGIN
BL:=ORD(BLK.CBL);
BLOKKRES:=FALSE;
N:=POCETPRVKU(BL,LINE);M:=-1;
WHILE N>0 DO BEGIN
 NPRVEK(BL,M,LINE);
 IF PLINE.X=MAXINT THEN BEGIN
   A:=PLINE.Y;
   NPRVEK(BL,M+1,LINE);
   X1:=PLINE.X;Y1:=PLINE.Y;
   NPRVEK(BL,M+2,LINE);
   X2:=PLINE.X;Y2:=PLINE.Y;
   M:=M+2;N:=N-2;
   END
  ELSE BEGIN
   X1:=X2;Y1:=Y2;X2:=PLINE.X;Y2:=PLINE.
Y;
   END;
 IF LINETL(X1+BLK.X,Y1+BLK.Y,X2+BLK.X,Y
2+BLK.Y,A) THEN BEGIN
     BLOKKRES:=TRUE;
     IF (GBL=BL) AND (KRES=FALSE) THEN
BEGIN
       PLINE.X:=PLINE.X+20000;ZMEN(BL,
M,LINE);END;
     END;
 N:=N-1;M:=M+1;
 END;
N:=POCETPRVKU(BL,ARC);X1:=BLK.X;Y1:=BLK
.Y;
IF N>0 THEN BEGIN
 FOR M:=1 TO N DO BEGIN
  NPRVEK(BL,M,ARC);
  WITH PARC DO BEGIN
   XC:=XC+X1;YC:=YC+Y1;
   XS:=XS+X1;YS:=YS+Y1;
   END;
  IF ARCKRES(PARC) THEN BEGIN
   BLOKKRES:=TRUE;NPRVEK(BL,M,ARC);
   IF (GBL=BL) AND (KRES=FALSE) THEN BE
GIN
    PARC.XC:=PARC.XC+20000;ZMEN(BL,M,AR
C);END;
   END;
  END;
 END;
N:=POCETPRVKU(BL,TEXT);
IF N>0 THEN BEGIN
 FOR M:=1 TO N DO BEGIN
  NPRVEK(BL,M,TEXT);
  WITH PTEXT DO BEGIN
   XP:=XP+X1;YP:=YP+Y1;END;
  IF TEXTKRES(PTEXT) THEN BEGIN
  BLOKKRES:=TRUE;NPRVEK(BL,M,TEXT);
  IF (GBL=BL) AND (KRES=FALSE) THEN BEG
IN
   PTEXT.XP:=PTEXT.XP+20000;ZMEN(BL,M,T
EXT);END;
  END;
  END;
 END;
N:=POCETPRVKU(BL,BLOK);
IF N>0 THEN BEGIN
 FOR M:=1 TO N DO BEGIN
  NPRVEK(BL,M,BLOK);
  WITH PBLOK DO BEGIN
   X:=X+X1;Y:=Y+Y1;END;
  IF BLOKKRES(PBLOK) THEN BEGIN
   BLOKKRES:=TRUE:NPRVEK(BL,M,BLOK);
   IF (GBL=BL) AND (KRES=FALSE) THEN BE
GIN
    PBLOK.X:=PBLOK.X+20000;ZMEN(BL,M,BL
OK);END;
   END;
  END;
 END;
END;
PROCEDURE KRESL1(BK:BLOKT;KR:BOOLEAN);
VAR RESS:BOOLEAN;
BEGIN
KRES:=KR;GBL:=ORD(BK.CBL);
RESS:=BLOKKRES(BK);
KRES:=TRUE;
END;
PROCEDURE PAUSE(N:INTEGER);
VAR M:INTEGER;
BEGIN
FOR M:=1 TO N DO
END;
PROCEDURE DRAW(X,Y,X1,Y1:INTEGER;O:INTE
GER);
BEGIN
DRFT;PLOT(X,Y,O);PLOT(X1,Y1,O);DOT;
END;
PROCEDURE KURZOR(X,Y:INTEGER);
BEGIN
 DRAW(X-2,Y,X+2,Y,9);DRAW(X,Y-2,X,Y+2,9
);
END;
FUNCTION POHYBKUR(XR,YR:INTEGER;KR:BOOL
EAN):CHAR;
VAR A:CHAR;
    TIME,R,N:INTEGER;
PROCEDURE KRIZW;
BEGIN
 KURZOR(XCUR,YCUR);
 DRAW(XCUR,1,XCUR,175,9);DRAW(1,YCUR,20
7,YCUR,9);
 IF KR THEN BEGIN
             DRAW(XR,YR,XR,YCUR,9);DRAW
(XR,YR,XCUR,YR,9);END;
END;

BEGIN
R:=0;TIME:=0;
REPEAT
 A:=INCH;
 IF A=CHR(0) THEN BEGIN
  REPEAT
   A:=INCH;TIME:=TIME+1;
   IF TIME>3500 THEN BEGIN
    KRIZW;TIME:=0;
    REPEAT
     A:=INCH;
    UNTIL A<>CHR(0);
    KRIZW;END;
  UNTIL A<>CHR(0);
  TIME:=0;R:=0;END;
 R:=R+RYCHLOST[(R DIV 20+1].P;
 IF A='O' THEN XCUR:=XCUR-RYCHLOST[(R D
IV 20+1].XRY;
 IF A='P' THEN XCUR:=XCUR+RYCHLOST[(R D
IV 20+1].XRY;
 IF A='Q' THEN YCUR:=YCUR+RYCHLOST[(R D
IV 20+1].YRY;
 IF A='A' THEN YCUR:=YCUR-RYCHLOST[(R D
IV 20+1].YRY;
 IF (XCUR<2) OR (XCUR>205) OR (YCUR<2)
OR (YCUR>173) THEN BEGIN
    XCUR:=XCURS;YCUR:=YCURS;END;
WRITE(CHR(21),CHR(0));AT(0,0);WRITE(((X
CUR-PANX*SCALE)/SCALE*0.125):6:3);
AT(0,10);WRITE(((YCUR-PANY*SCALE)/SCALE
*0.125):6:3);
 KURZOR(XCURS,YCURS);KURZOR(XCUR,YCUR);
 XCURS:=XCUR;YCURS:=YCUR;
 PAUSE(900);
UNTIL NOT((A='O') OR (A='P') OR (A='Q')
 OR (A='A'));
POHYBKUR:=A;WRITE(CHR(21),CHR(0));
REPEAT UNTIL INCH=CHR(0);
END;
PROCEDURE ZOBRAZCAST(C:INTEGER);FORWARD
;
PROCEDURE OBNOVNAPIS;FORWARD;
PROCEDURE ZOBRAZCAST;
VAR N:INTEGER;
BEGIN
WRITE(CHR(21),CHR(0));
FOR N:=C TO C+16 DO BEGIN
 AT(N-C+2,26);
 IF N>PBLAKT+POCPRIK THEN WRITE('
')
                       ELSE WRITE(JMENA[N
]);
 END;
END;
PROCEDURE INVERSE(C:INTEGER;SW:BOOLEAN)
;
BEGIN
AT(C,26);
IF SW THEN WRITE(CHR(17),CHR(2))
      ELSE WRITE(CHR(17),CHR(0));
WRITE(CHR(21),CHR(1),'      ');
WRITE(CHR(21),CHR(0),CHR(17),CHR(0));
END;
FUNCTION FCEMNU(B:INTEGER):CHAR;
VAR A:CHAR;
BEGIN
IF MNUA<B THEN BEGIN
 MNUA:=B;ZOBRAZCAST(MNUA);END;
INVERSE(MNUR+2,TRUE);
REPEAT UNTIL INCH=CHR(0);
REPEAT
 REPEAT
  A:=INCH;
 UNTIL A<>CHR(0);
 IF A='O' THEN BEGIN
  MNUA:=MNUA-10;
  IF MNUA<B THEN MNUA:=B;

  ZOBRAZCAST(MNUA);INVERSE(MNUR+2,TRUE)
;END;
 IF A='P' THEN BEGIN
  MNUA:=MNUA+10;
  IF MNUA>(PBLAKT+POCPRIK-16) THEN MNUA
:=PBLAKT+POCPRIK-16;
  ZOBRAZCAST(MNUA);INVERSE(MNUR+2,TRUE)
;END;
 IF A='Q' THEN BEGIN
  MNUR:=MNUR-1;
  IF MNUR<0 THEN MNUR:=0;
  INVERSE(MNUR+3,FALSE);INVERSE(MNUR+2,
TRUE);END;
 IF A='A' THEN BEGIN
  MNUR:=MNUR+1;
  IF MNUR>16 THEN MNUR:=16;
  INVERSE(MNUR+1,FALSE);INVERSE(MNUR+2,
TRUE);END;
 PAUSE(900);
UNTIL NOT((A='O') OR (A='P') OR (A='Q')
 OR (A='A'));
FCEMNU:=A;INVERSE(MNUR+2,FALSE);
REPEAT UNTIL INCH=CHR(0);
IF (MNUA+MNUR<B) OR (MNUA+MNUR>POCPRIK+
PBLAKT) THEN BEGIN
MNUA:=B;MNUR:=0;ZOBRAZCAST(B);END;
END;
PROCEDURE OBNOVNAPIS;
BEGIN
AT(22,0);WRITE(JMENA[DP[1]],' MENU   ',
JMENA[DP[2]],' ',JMENA[DP(3)]);
END;
FUNCTION CPRIK(A:CHAR):INTEGER;
BEGIN
CPRIK:=0;
IF A='X' THEN BEGIN
 AT(22,0);WRITE('SELECT CREATE DEFINE S
ELECT');
 A:=FCEMNU(1);
```

```
 IF (A='Z') OR (A='M') THEN CPRIK:=MNUA
+MNUR;
 IF A='N' THEN BEGIN
  AT(22,0);WRITE('Select key (Z,N,M)
      ');
  REPEAT UNTIL INCH<>CHR(0);
  A:=INCH;
  IF A='Z' THEN DP[1]:=MNUA+MNUR;
  IF A='N' THEN DP[2]:=MNUA+MNUR;
  IF A='M' THEN DP[3]:=MNUA+MNUR;
  REPEAT UNTIL INCH=CHR(0);
  END;
 OBNOVNAPIS;END
 ELSE BEGIN
  IF A='Z' THEN CPRIK:=DP[1];
  IF A='N' THEN CPRIK:=DP[2];
  IF A='M' THEN CPRIK:=DP[3];
  END;
END;

PROCEDURE INITIAL;.
PROCEDURE INITDISP;
BEGIN
 PANX:=500;PANY:=400;SCALE:=1/5;KRES:=T
RUE;
 X3:=0;Y3:=0;X4:=100;Y4:=100;
END;
PROCEDURE INITDM;
VAR N:INTEGER;POBJ:OBJT;
BEGIN
FOR N:=1 TO PBLMAX DO BEGIN
 FOR POBJ:=LINE TO BLOK DO BEGIN
UK[N].PRVKY[POBJ]:=OFFSET;
UK[N].POCET[POBJ]:=0;
END;
END;
ENDOF:=OFFSET;
END;
PROCEDURE INITMNU;
BEGIN
XCUR:=100;XCURS:=100;YCUR:=80;YCURS:=80
;
RYCHLOST[1].XRY:=1;RYCHLOST[1].YRY:=1
;RYCHLOST[1].P:=4;
RYCHLOST[2].XRY:=2;RYCHLOST[2].YRY:=2
;RYCHLOST[2].P:=2;
RYCHLOST[3].XRY:=7;RYCHLOST[3].YRY:=7
;RYCHLOST[3].P:=0;
PAGE;KURZOR(XCUR,YCUR);
PBLAKT:=0;CURBLOK:=1;CURLT:=101;
FOR GBL:=1 TO JMCELKEM DO JMENA[GBL]:='
    ';
JMENA[1]:='OPEN  ';JMENA[2]:='CURR  ';J
MENA[5]:='MOVE  ';
JMENA[6]:='COPY  ';JMENA[7]:='ERASE ';J
MENA[8]:='L.TYPE';
JMENA[11]:='ZOOM  ';JMENA[12]:='PAN   '
;
JMENA[13]:='CURSOR';JMENA[15]:='SAVE  '
;
JMENA[16]:='LOAD  ';JMENA[19]:='LINE  '
;
JMENA[20]:='ARC   ';JMENA[21]:='TEXT  '
;
JMENA[24]:='blok 1';
JMENA[18]:=' def18';JMENA[22]:=' def22'
;
POKE(23659,CHR(1));
MNUA:=1;MNUR:=15;ZOBRAZCAST(1);
DP[1]:=7;DP[2]:=21;DP[3]:=19;
OBNOVNAPIS;
END;
BEGIN
 INITDISP;
 INITDM;
 INITMNU;
END;
PROCEDURE KUR;
BEGIN KURZOR(XCUR,YCUR);WRITE(CHR(21),C
HR(0));END;

FUNCTION XSK:INTEGER;
BEGIN
XSK:=ROUND((XCUR-PANX*SCALE)/SCALE);END
;
FUNCTION YSK:INTEGER;
BEGIN
YSK:=ROUND((YCUR-PANY*SCALE)/SCALE);END
;
PROCEDURE PRLINE;
VAR A:CHAR;
    XM,YM:INTEGER;
    B:BOOLEAN;
BEGIN
PLINE.X:=MAXINT;PLINE.Y:=CURLT;VLOZ(CUR
BLOK,LINE);
PLINE.X:=XSK;PLINE.Y:=YSK;XM:=XSK;YM:=Y
SK;VLOZ(CURBLOK,LINE);
AT(22,0);WRITE('To point
      ');
A:=POHYBKUR(0,0,FALSE);
PLINE.X:=XSK;PLINE.Y:=YSK;
KUR;
B:=LINETL(XM,YM,XSK,YSK,CURLT);VLOZ(CUR
BLOK,LINE);
KUR;
XM:=XSK;YM:=YSK;
AT(22,0);WRITE('END   Next point ');
WHILE NOT(POHYBKUR(0,0,FALSE)='Z') DO B
EGIN
 PLINE.X:=XSK;PLINE.Y:=YSK;VLOZ(CURBLOK
,LINE);
KUR;
 B:=LINETL(XM,YM,XSK,YSK,CURLT);XM:=XSK
;YM:=YSK;
KUR;
 END;
OBNOVNAPIS;
END;
PROCEDURE PRARC;
VAR F:CHAR;B:BOOLEAN;
BEGIN
AT(22,0);WRITE('Center point
      ');
F:=POHYBKUR(0,0,FALSE);
PARC.XC:=XSK;PARC.YC:=YSK;
AT(22,0);WRITE('Start point  ');
F:=POHYBKUR(0,0,FALSE);
PARC.XS:=XSK;PARC.YS:=YSK;
AT(22,0);WRITE('Enter angle  ');
AT(22,13);READLN;READ(PARC.U);
PARC.U:=ROUND(PARC.U/180*3.1415*1000);
PARC.A:=CURLT;
 KUR;
VLOZ(CURBLOK,ARC);B:=ARCKRES(PARC);OBNO
VNAPIS;
 KUR;
END;
PROCEDURE PRTEXT;
VAR B:BOOLEAN;
    C:REAL;

BEGIN
AT(22,0);WRITE('Enter text
      ');
AT(22,12);READLN;READ(PTEXT.T);
AT(22,0);WRITE('Size          mm
      ');
AT(22,5);READLN;READ(C);
PTEXT.VELIKOST:=CHR(ROUND(ABS(C)*2));
PTEXT.XP:=XSK;PTEXT.YP:=YSK;VLOZ(CURBLO
K,TEXT);
 KUR;
B:=TEXTKRES(PTEXT);
 KUR;
OBNOVNAPIS;
END;
PROCEDURE PRBLOK(A:INTEGER);
VAR B:BOOLEAN;
BEGIN
IF (A>POCPRIK) AND (A<=POCPRIK+PBLAKT)
AND (A<>CURBLOK+POCPRIK) THEN BEGIN
 PBLOK.X:=XSK;PBLOK.Y:=YSK;PBLOK.CBL:=C
HR(A-POCPRIK);
KUR;
 VLOZ(CURBLOK,BLOK);B:=BLOKKRES(PBLOK);
KUR;
 END;
END;
PROCEDURE CLWA;
VAR N:INTEGER;
BEGIN
 WRITE(CHR(21),CHR(0));
 FOR N:=0 TO 21 DO BEGIN
  AT(N,0);WRITE('
    ');
  END;
END;
PROCEDURE REDRAW;
BEGIN
 PBLOK.X:=0;PBLOK.Y:=0;PBLOK.CBL:=CHR(C
URBLOK);
 KRESLI(PBLOK,TRUE);
 KUR;
END;
PROCEDURE PRCURR;
VAR A:CHAR;
    N,M:INTEGER;
BEGIN
AT(22,0);WRITE('Backgr.  Current block
      ');
CLWA;
REPEAT
 A:=FCEMNU(POCPRIK+1);
 IF A='Z' THEN BEGIN
      CURBLOK:=MNUA+MNUR-POCPRIK;
      REDRAW;
      FOR N:=0 TO 103 DO BEGIN
       FOR M:=0 TO 87 DO BEGIN
        PLOT(N*2,M*2,0);PLOT(N*2,M*2,9)
;
        PLOT(N*2+1,M*2+1,0);PLOT(N*2+1,
M*2+1,9);
       END;
      END;
     END
   ELSE BEGIN
      CURBLOK:=MNUA+MNUR-POCPRIK;REDRAW
;END;
UNTIL NOT(A='Z');
OBNOVNAPIS;
END;
PROCEDURE PROPEN;
BEGIN
IF PBLAKT=PBLMAX THEN BEGIN
      AT(22,0);WRITE('Too many blocks
        ');
      PAUSE(30000);END
   ELSE BEGIN
      PBLAKT:=PBLAKT+1;CURBLOK:=PBLAKT;
      AT(22,0);WRITE('Name
        ');
      AT(22,6);READLN;READ(JMENA[POCPRI
K+CURBLOK]);
      ZOBRAZCAST(PBLAKT+POCPRIK-15);CLW
A;
      END;
OBNOVNAPIS;KUR;
END;
PROCEDURE WINDOW;FORWARD;
PROCEDURE ZJISTIL(BL,P:INTEGER;VAR N,M
:INTEGER);
VAR O:INTEGER;
BEGIN
N:=P;
REPEAT
 N:=N-1;NPRVEK(BL,N,LINE);
UNTIL PLINE.X=MAXINT;
M:=P;O:=POCETPRVKU(BL,LINE);
REPEAT
 M:=M+1;NPRVEK(BL,M,LINE);
UNTIL (PLINE.X=MAXINT) OR (M=O+1);
M:=M-1;
END;
PROCEDURE ERASE(BL:INTEGER);
VAR M,N,O:INTEGER;
    OBJ:OBJT;
BEGIN
WINDOW;
M:=1;N:=POCETPRVKU(BL,LINE);
WHILE N>0 DO BEGIN
 NPRVEK(BL,M,LINE);
 IF (PLINE.X>10000) AND (PLINE.X<MAXINT
) THEN BEGIN
   ZJISTIL(BL,M,N,O);
   FOR M:=N TO O DO SMAZ(BL,N,LINE);
   N:=POCETPRVKU(BL,LINE)+1;M:=0;
   END;
 N:=N-1;M:=M+1;
 END;
FOR OBJ:=ARC TO BLOK DO BEGIN
 M:=1;
 FOR N:=1 TO POCETPRVKU(BL,OBJ) DO BEGI
N
  NPRVEK(BL,M,OBJ);
  CASE OBJ OF
   ARC:O:=PARC.XC;
   TEXT:O:=PTEXT.XP;
   BLOK:O:=PBLOK.X
  END;
  IF O>10000 THEN BEGIN
   SMAZ(BL,M,OBJ);M:=M-1;END;
  M:=M+1;
  END;
 END;
 CLWA;REDRAW;OBNOVNAPIS;
END;
```

```
PROCEDURE WINDOW;
VAR A:CHAR;
    XR,YR:INTEGER;
BEGIN
AT(22,0);WRITE('Left down corner
          ');
A:=POHYBKUR(0,0,FALSE);
X3:=XCUR;Y3:=YCUR;XR:=XCUR;YR:=YCUR;
AT(22,0);WRITE('Right up corner   ');
A:=POHYBKUR(XR,YR,TRUE);
X4:=XCUR;Y4:=YCUR;
PBLOK.X:=0;PBLOK.Y:=0;PBLOK.CBL:=CHR(CU
RBLOK);
KRESLI(PBLOK,FALSE);
END;
PROCEDURE PMCZAC;
VAR A:CHAR;
    XR,YR:INTEGER;
BEGIN
WINDOW;
AT(22,0);WRITE('New left down corner
          ');
A:=POHYBKUR(0,0,FALSE);
X4:=ROUND((XCUR-PANX*SCALE)/SCALE)-ROUN
D((X3-PANX*SCALE)/SCALE);
Y4:=ROUND((YCUR-PANY*SCALE)/SCALE)-ROUN
D((Y3-PANY*SCALE)/SCALE);
END;
PROCEDURE MOVECOPY(MOVE:BOOLEAN;BL,DX,D
Y:INTEGER);
VAR M,N,O,P,Q:INTEGER;
       OBJ:OBJT;
BEGIN
Q:=POCETPRVKU(BL,LINE);
FOR N:=1 TO Q DO BEGIN
 NPRVEK(BL,N,LINE);
 IF (PLINE.X>10000) AND (PLINE.X<MAXINT
) THEN BEGIN
  ZJISTILI(BL,N,M,O);
  IF MOVE=FALSE THEN BEGIN NPRVEK(BL,M,
LINE);VLOZ(BL,LINE);END;
  FOR P:=M+1 TO O DO BEGIN
   NPRVEK(BL,P,LINE);
   IF PLINE.X>10000 THEN PLINE.X:=PLINE
.X-20000;
   ZMEN(BL,P,LINE);
   PLINE.X:=PLINE.X+DX;PLINE.Y:=PLINE.Y
+DY;
   IF MOVE THEN ZMEN(BL,P,LINE)
           ELSE VLOZ(BL,LINE);
  END;
 END;
END;
FOR OBJ:=ARC TO BLOK DO BEGIN
 Q:=POCETPRVKU(BL,OBJ);
 FOR N:=1 TO Q DO BEGIN
  NPRVEK(BL,N,OBJ);
  CASE OBJ OF
    ARC:O:=PARC.XC;
   TEXT:O:=PTEXT.XP;
   BLOK:O:=PBLOK.X
  END;
  IF O>10000 THEN BEGIN
   CASE OBJ OF
     ARC:BEGIN
          PARC.XC:=PARC.XC-20000;ZMEN(B
L,N,OBJ);
          WITH PARC DO BEGIN
           XC:=XC+DX;YC:=YC+DY;
           XS:=XS+DX;YS:=YS+DY;
           END;
          END;
    TEXT:BEGIN
          PTEXT.XP:=PTEXT.XP-20000;ZMEN
(BL,N,OBJ);
          PTEXT.XP:=PTEXT.XP+DX;PTEXT.Y
P:=PTEXT.YP+DY;
          END;
    BLOK:BEGIN
          PBLOK.X:=PBLOK.X-20000;ZMEN(B
L,N,OBJ);
          PBLOK.X:=PBLOK.X+DX;PBLOK.Y:=
PBLOK.Y+DY;
          END
    END;
   IF MOVE THEN ZMEN(BL,N,OBJ)
           ELSE VLOZ(BL,OBJ);
  END;
 END;
 CLWA;REDRAW;OBNOVNAPIS;
END;

PROCEDURE VYPAN(ZPX,ZPY:REAL);FORWARD;
PROCEDURE PRLINETYPE;
VAR A:INTEGER;
BEGIN
AT(22,0);WRITE('Number of pereel lines
          ');
AT(22,24);READLN;READ(A);
IF (A>99) OR (A<1) THEN A:=1;
CURLT:=A;
AT(22,0);WRITE('Distance between lines
          ');
AT(22,24);READLN;READ(A);
IF (A>99) OR (A<0) THEN A:=1;
CURLT:=CURLT+A*100;
OBNOVNAPIS;
END;
PROCEDURE PRPAN;
VAR ZPX,ZPY:REAL;
BEGIN
AT(22,0);WRITE('Enter X coord.
          ');
AT(22,17);READLN;READ(ZPX);
AT(22,0);WRITE('Enter Y coord.
          ');
AT(22,17);READLN;READ(ZPY);
ZPX:=-ZPX;ZPY:=-ZPY;
VYPAN(ZPX,ZPY);
CLWA;REDRAW;OBNOVNAPIS;
END;
PROCEDURE PRZOOM;
VAR SC,ZX,ZY:REAL;
BEGIN
REPEAT
 AT(22,0);WRITE('Enter scale 1:
          ');
 AT(22,14);READLN;READ(SC);
UNTIL (SC>=1) AND (SC<100);
SC:=1/SC;
ZX:=(PANX-100/SCALE)/8;
ZY:=(PANY-80/SCALE)/8;
SCALE:=SC;
VYPAN(ZX,ZY);
CLWA;REDRAW;OBNOVNAPIS;
END;
PROCEDURE VYPAN;
BEGIN
 PANX:=ROUND(ZPX*8+100/SCALE);
 PANY:=ROUND(ZPY*8+80/SCALE);
END;
PROCEDURE PRSAVE;
VAR A:CHAR;
    OD,DOB:INTEGER;
    NAME:ARRAY[1..8] OF CHAR;
BEGIN
NAME:='        ';
AT(22,0);WRITE('First block
          ');
A:=FCEMNU(POCPRIK+1);
OD:=MNUA+MNUR;
AT(22,0);WRITE(' Last');
A:=FCEMNU(OD);
DOB:=MNUA+MNUR;
AT(22,0);WRITE('Name          ');
AT(22,5);READLN;READ(NAME);
NAME[6]:='.';NAME[7]:='J';NAME[8]:='M';
TOUT(NAME,ADDR(JMENA)+6*(OD-1),6*(DOB-O
D+1));
OD:=OD-POCPR)K;DOB:=DOB-POCPR)K;
NAME[7]:='U';NAME[8]:='K';
TOUT(NAME,ADDR(UK)+16*(OD-1),16*(DOB-OD
+1));

NAME[7]:='V';NAME[8]:='S';
TOUT(NAME,UK[OD].PRVKY[LINE],UK[DOB+1].
PRVKY[LINE]-UK[OD].PRVKY[LINE]);
PAGE;
ZOBRAZCAST(MNUA);OBNOVNAPIS;
PBLOK.X:=0;PBLOK.Y:=0;PBLOK.CBL:=CHR(CU
RBLOK);
KRESLI(PBLOK,TRUE);KURZOR(XCUR,YCUR);
END;
PROCEDURE PRLOAD;
VAR N,M,DIF:INTEGER;
    NAME:ARRAY[1..8] OF CHAR;
    POBJ:OBJT;
BEGIN
NAME:='        ';
AT(22,0);WRITE('Name
          ');
AT(22,5);READLN;READ(NAME);
NAME[6]:='.';NAME[7]:='J';NAME[8]:='M';
AT(22,0);
TIN(NAME,ADDR(JMENA)+6*(POCPRIK+PBLAKT)
);
N:=PBLAKT+POCPRIK+1;
WHILE (N<=JMCELKEM) AND (JMENA[N]<>'
   ') DO N:=N+1;
N:=N-1-POCPRIK;
NAME[7]:='U';NAME[8]:='K';
TIN(NAME,ADDR(UK)+16*PBLAKT);
DIF:=UK[PBLAKT+1].PRVKY[LINE]-ENDOF;
IF UK[N].PRVKY[BLOK]+5*UK[N].POCET[BLOK
]-DIF<TOPMEM THEN BEGIN
FOR M:=PBLAKT+1 TO N DO BEGIN
 FOR POBJ:=LINE TO BLOK DO
  UK[M].PRVKY[POBJ]:=UK[M].PRVKY[POBJ]-
D)F;
 END;
DIF:=UK[N].PRVKY[BLOK]+5*UK[N].POCET[BL
OK];
IF N<PBLMAX THEN BEGIN
 FOR M:=N+1 TO PBLMAX DO BEGIN
  FOR POBJ:=LINE TO BLOK DO UK[M].PRVKY
[POBJ]:=DIF;
  END;
 END;
NAME[7]:='V';NAME[8]:='S';
TIN(NAME,ENDOF);
PBLAKT:=N;ENDOF:=DIF;
END
ELSE BEGIN
 AT(22,0);WRITE('No memory for file
          ');
 PAUSE(30000);END;
PAGE;
ZOBRAZCAST(MNUA);OBNOVNAPIS;
PBLOK.X:=0;PBLOK.Y:=0;PBLOK.CBL:=CHR(CU
RBLOK);
KRESLI(PBLOK,TRUE);KURZOR(XCUR,YCUR);
END;
BEGIN
 INITIAL;
 REPEAT
  CISLO:=CPRIK(POHYBKUR(0,0,FALSE));
IF CISLO=19 THEN PRLINE;
IF CISLO=20 THEN PRARC;
IF CISLO=21 THEN PRTEXT;
IF CISLO>POCPRIK THEN PRBLOK(CISLO);
IF CISLO=1 THEN PROPEN;
IF CISLO=2 THEN PRCURR;
IF CISLO=7 THEN ERASE(CURBLOK);
IF CISLO=5 THEN BEGIN
                PMCZAC;
                MOVECOPY(TRUE,CURBLOK
,X4,Y4);END;
IF CISLO=6 THEN BEGIN
                PMCZAC;
                MOVECOPY(FALSE,CURBLO
K,X4,Y4);END;
IF CISLO=8 THEN PRLINETYPE;
IF CISLO=11 THEN PRZOOM;
IF CISLO=12 THEN PRPAN;
)F CISLO=15 THEN PRSAVE;
IF CISLO=16 THEN PRLOAD;
 UNTIL FALSE;
END.
```

Výpis 2. Program ZCF (929–V2)

```
PROGRAM A;
CONST PBLMAX=60;
      POCPRIK=23;
      JMCELKEM=83;
      OFFSET=$C350;TOPMEM=$EA00;
TYPE LINET=RECORD
          X,Y:INTEGER;END;
     ARCT=RECORD XC,YC,XS,YS,A,U:INTEGE
R;END;
     TEXTT=RECORD VELIKOST:CHAR;XP,YP:I
NTEGER;T:ARRAY[1..5] OF CHAR;END;
      OBJT=(LINE,ARC,TEXT,BLOK);
      UKAZATELE=ARRAY[1..PBLMAX] OF RE
CORD
                                      P
RVKY:ARRAY[LINE..BLOK] OF INTEGER;
                                      P
OCET:ARRAY[LINE..BLOK] OF INTEGER;
                                      E
ND;
     BLOKT=RECORD X,Y:INTEGER;CBL:CHAR;
END;
```

```
      NAMET=ARRAY[1..10] OF CHAR;
      STRING=ARRAY[1..100] OF CHAR;
      MALYSTRING=ARRAY[1..5] OF CHAR;
VAR PLINE:LINET;PARC:ARCT;PTEXT:TEXTT;P
BLOK:BLOKT;
    PBLAKT:INTEGER;
    UK:UKAZATELE;ENDOF:INTEGER;
    JMENA:ARRAY[1..JMCELKEM,1..6] OF CH
AR;
    CISRET:STRING;
PROCEDURE CISPREV(C:REAL);
VAR A,B:REAL;N:INTEGER;
FUNCTION POMC(B:REAL):CHAR;
BEGIN
POMC:=CHR(ROUND((A-(ENTIER(A/B/10)*10*B
+FRAC(A/B)*B))/B)+48);
END;
BEGIN
A:=ABS(C);B:=1000;CISRET:='          ';
IF C<0 THEN CISRET[1]:='-';
FOR N:=3 DOWNTO 0 DO BEGIN CISRET[5-N]:
=POMC(B);B:=B/10;END;
CISRET[6]:='.';
FOR N:=-1 DOWNTO -2 DO BEGIN CISRET[6-N
]:=POMC(B);B:=B/10;END;
END;
PROCEDURE ODVYS4(T:STRING);
VAR N:INTEGER;
BEGIN
FOR N:=1 TO 10 DO BEGIN
 POKE(16384,T[N]);
 INLINE(*3A,0,*40,*CD,*5D,*EC);END;
END;
PROCEDURE ODVYS5(T:STRING);
VAR N:INTEGER;
BEGIN
T[10]:=CHR(13);
FOR N:=1 TO 10 DO BEGIN
 POKE(16384,T[N]);
 INLINE(*3A,0,*40,*CD,*60,*EC);END;
END;
PROCEDURE ODVENT;
VAR S:STRING;
BEGIN
S:='          ';
S[10]:=CHR(13);ODVYS4(S);
END;
PROCEDURE ODVCAR;
BEGIN
ODVYS4(',       ');
END;
PROCEDURE MA(X,Y:REAL);
BEGIN
ODVYS4('MA      ');
CISPREV(X);ODVYS4(CISRET);
ODVCAR;
CISPREV(Y);ODVYS4(CISRET);
ODVENT;
END;
PROCEDURE VA(X,Y:REAL);
BEGIN
ODVYS4('VA      ');
CISPREV(X);ODVYS4(CISRET);
ODVCAR;
CISPREV(Y);ODVYS4(CISRET);
ODVENT;
END;
PROCEDURE OG(X,Y:REAL);
BEGIN
ODVYS4('OG      ');
CISPREV(X);ODVYS4(CISRET);ODVCAR;CISPRE
V(Y);ODVYS4(CISRET);ODVENT;
END;
PROCEDURE MF;
BEGIN ODVYS4('MF      ');ODVENT;END;
PROCEDURE SC(S:REAL);
BEGIN
ODVYS4('SC      ');CISPREV(S);ODVYS4(
CISRET);ODVENT;END;
PROCEDURE LT(L:INTEGER);
BEGIN ODVYS4('LT      ');CISPREV(L);O
DVYS4(CISRET);END;
PROCEDURE CS(X,Y:REAL);
BEGIN
ODVYS4('CS      ');
CISPREV(X);ODVYS4(CISRET);ODVCAR;CISPRE
V(Y);ODVYS4(CISRET);
ODVYS4('       ');ODVENT;
END;
PROCEDURE AC(A,B,C:REAL);
BEGIN
ODVYS4('AC      ');
CISPREV(A);ODVYS4(CISRET);ODVCAR;
CISPREV(B);ODVYS4(CISRET);ODVCAR;
CISPREV(C);ODVYS4(CISRET);ODVENT;
END;
PROCEDURE SPEED(S:INTEGER);
BEGIN
POKE(*EBAD,S);END;
PROCEDURE LINETL(X1,Y1,X2,Y2,A:INTEGER)
;
VAR N,M,MFX,MFY:INTEGER;M1,M2,XR,YR:REA
L;
BEGIN
LT(0);
IF A=2 THEN BEGIN A:=1;LT(3);END;
IF A=3 THEN BEGIN A:=1;LT(2);END;
N:=A MOD 100; A:=A DIV 100;
M1:=SQRT(SQR(X2-X1)+SQR(Y2-Y1));
IF M1=0 THEN M1:=1;
M2:=(Y2-Y1)/M1*A*(N-1)/2;
M1:=(X2-X1)/M1*A*(N-1)/2;
X2:=X2-X1;XR:=X1-M2;
Y2:=Y2-Y1;YR:=Y1+M1;
IF N>1 THEN BEGIN M1:=M1/((N-1)/2);M2:=
M2/((N-1)/2);END;
FOR M:=1 TO N DO BEGIN
X1:=ROUND(XR);Y1:=ROUND(YR);
MA(X1,Y1);
VA(X1+X2,Y1+Y2);
XR:=XR+M2;YR:=YR-M1;
END;
END;
PROCEDURE ARCTL(XC,YC,XS,YS,U,A:INTEGER
);
VAR POCUHEL,KONUHEL,PR,PS:REAL;
    M,CAR,VZD:INTEGER;
BEGIN
LT(0);
IF A=2 THEN BEGIN A:=1;LT(3);END;
IF A=3 THEN BEGIN A:=1;LT(2);END;
CAR:=A MOD 100;VZD:=A DIV 100;
PR:=SQRT(SQR(XS-XC)+SQR(YS-YC));
IF U<0 THEN PR:=-PR;
POCUHEL:=ARCTAN((YS-YC)/(XS-XC+0.01));
IF (XS-XC)<0 THEN POCUHEL:=POCUHEL+3.14
159;
KONUHEL:=POCUHEL+U/1000;
IF KONUHEL>6.2832 THEN KONUHEL:=KONUHEL
-6.2832;
IF ABS(U)>6196 THEN KONUHEL:=POCUHEL;
PS:=PR-(CAR-1)/2*VZD;
FOR M:=1 TO CAR DO BEGIN
MA(XC,YC);
AC(PS,POCUHEL,KONUHEL);
PS:=PS+VZD;
END;
END;
PROCEDURE TEXTKRES(VELIKOST:CHAR;XP,YP:
INTEGER;T:MALYSTRING);
VAR N:INTEGER;
    VEL:REAL;
    T1:ARRAY[1..10] OF CHAR;
BEGIN
VEL:=ORD(VELIKOST)/2*8;
T1:='          ';
MA(XP,YP);
CS(VEL,VEL);
FOR N:=1 TO 5 DO T1[N]:=T[N];
T1[6]:=CHR(13);
ODVYS5(T1);
END;
PROCEDURE LDIR(HL,DE,BC:INTEGER);
BEGIN
IF BC>0 THEN BEGIN
POKE (23600,HL);POKE (23602,DE);POKE (2
3604,BC);
INLINE (*2A,*30,*5C,*ED,*5B,*32,*5C,*ED
,*4B,*34,*5C);
INLINE (*ED,*B0);
END;
END;
PROCEDURE LDDR(HL,DE,BC:INTEGER);
BEGIN
IF BC>0 THEN BEGIN
POKE(23600,HL);POKE(23602,DE);POKE(2360
4,BC);
INLINE (*2A,*30,*5C,*ED,*5B,*32,*5C,*ED
,*4B,*34,*5C);
INLINE (*ED,*B8);
END;
END;
FUNCTION POCETPRVKU(BLOK:INTEGER;OBJ:OB
JT):INTEGER;
BEGIN
 POCETPRVKU:=UK[BLOK].POCET[OBJ];
END;
PROCEDURE POINTERS(BL,M:INTEGER;OBJ:OBJ
T);
VAR N:INTEGER;
    POBJ:OBJT;
BEGIN
IF OBJ<BLOK THEN BEGIN
 FOR POBJ:=SUCC(OBJ) TO BLOK DO
  UK[BL].PRVKY[POBJ]:=UK[BL].PRVKY[POBJ
]+M;
END;
IF BL<PBLMAX THEN BEGIN
 FOR N:=BL+1 TO PBLMAX DO BEGIN
  FOR POBJ:=LINE TO BLOK DO
   UK[N].PRVKY[POBJ]:=UK[N].PRVKY[POBJ]
+M;
 END;
END;
ENDOF:=ENDOF+M;
END;
PROCEDURE ADLEN(OBJ:OBJT;VAR N,M:INTEGE
R);
BEGIN
CASE OBJ OF
 LINE:BEGIN N:=ADDR(PLINE);M:=4;END;
 ARC:BEGIN N:=ADDR(PARC);M:=12;END;
 TEXT:BEGIN N:=ADDR(PTEXT);M:=10;END;
 BLOK:BEGIN N:=ADDR(PBLOK);M:=5;END
END;
END;
PROCEDURE NPRVEK(BL,O:INTEGER;OBJ:OBJT)
;

VAR N,M:INTEGER;
BEGIN
ADLEN(OBJ,N,M);
 LDIR((O-1)*M+UK[BL].PRVKY[OBJ],N,M);
END;
PROCEDURE BLOKKRES(BLK:BLOKT);
VAR X1,Y1,X2,Y2,M,N,A1P,BL:INTEGER;
BEGIN
BL:=ORD(BLK.CBL);
N:=POCETPRVKU(BL,LINE);M:=1;
WHILE N>0 DO BEGIN
 NPRVEK(BL,M,LINE);
 IF PLINE.X=MAXINT THEN BEGIN
   A1P:=PLINE.Y;
   NPRVEK(BL,M+1,LINE);
   X1:=PLINE.X;Y1:=PLINE.Y;
   NPRVEK(BL,M+2,LINE);
   X2:=PLINE.X;Y2:=PLINE.Y;
   M:=M+2;N:=N-2;
   END
  ELSE BEGIN
   X1:=X2;Y1:=Y2;X2:=PLINE.X;Y2:=PLINE.
Y;
   END;
    LINETL(X1+BLK.X,Y1+BLK.Y,X2+BLK.X,Y
2+BLK.Y,A1P);
 N:=N-1;M:=M+1;
 END;
N:=POCETPRVKU(BL,ARC);X1:=BLK.X;Y1:=BLK
.Y;
IF N>0 THEN BEGIN
 FOR M:=1 TO N DO BEGIN
  NPRVEK(BL,M,ARC);
  WITH PARC DO BEGIN
   XC:=XC+X1;YC:=YC+Y1;
   XS:=XS+X1;YS:=YS+Y1;
   ARCTL(XC,YC,XS,YS,U,A);
   END;
  END;
 END;
N:=POCETPRVKU(BL,TEXT);
IF N>0 THEN BEGIN
 FOR M:=1 TO N DO BEGIN
  NPRVEK(BL,M,TEXT);
  WITH PTEXT DO BEGIN
   XP:=XP+X1;YP:=YP+Y1;
   TEXTKRES(VELIKOST,XP,YP,T);END;
  END;
 END;
N:=POCETPRVKU(BL,BLOK);
IF N>0 THEN BEGIN
 FOR M:=1 TO N DO BEGIN
  NPRVEK(BL,M,BLOK);
  WITH PBLOK DO BEGIN
   X:=X+X1;Y:=Y+Y1;END;
  BLOKKRES(PBLOK);
  END;
```

```
 END;
END;
PROCEDURE PRLOAD;
VAR N,M,DIF:INTEGER;
    NAME:ARRAY[1..8] OF CHAR;
    POBJ:OBJT;
BEGIN
NAME:='        ';
AT(20,0];WRITE('Name
       ');
AT(20,5);READLN;READ(NAME);
NAME[6]:='.';NAME[7]:='J';NAME[8]:='M';
TIN(NAME,ADDR(JMENA)+6*(POCPRIK+PBLAKT)
);
N:=PBLAKT+POCPRIK+1;
WHILE (N<=JMCELKEM] AND (JMENA[N]<>'
   ') DO N:=N+1;
N:=N-1-POCPRIK;
NAME[7]:='U';NAME[8]:='K';
TIN(NAME,ADDR(UK)+16*PBLAKT);
DIF:=UK[PBLAKT+1].PRVKY[LINE]-ENDOF;
IF UK[N].PRVKY[BLOK]+5*UK[N].POCET[BLOK
]-DIF<TOPMEM THEN BEGIN
FOR M:=PBLAKT+1 TO N DO BEGIN
 FOR POBJ:=LINE TO BLOK DO
  UK[M].PRVKY[POBJ]:=UK[M].PRVKY[POBJ]-
DIF;
 END;
DIF:=UK[N].PRVKY[BLOK]+5*UK[N].POCET[BL
OK];
IF N<PBLMAX THEN BEGIN
 FOR M:=N+1 TO PBLMAX DO BEGIN
  FOR POBJ:=LINE TO BLOK DO UK[M].PRVKY
[POBJ]:=DIF;
  END;
 END;
NAME[7]:='V';NAME[8]:='S';
TIN(NAME,ENDOF);
PBLAKT:=N;ENDOF:=DIF;
END
ELSE BEGIN
 AT(20,0];WRITE('No memory for file
       ');
END;
END;
PROCEDURE INIT;
VAR N:INTEGER;POBJ:OBJT;
BEGIN
FOR N:=1 TO PBLMAX DO BEGIN
 FOR POBJ:=LINE TO BLOK DO BEGIN
  UK[N].PRVKY[POBJ]:=OFFSET;
  UK[N].POCET[POBJ]:=0;
 END;
END;
ENDOF:=OFFSET;
FOR N:=1 TO JMCELKEM DO JMENA[N]:='
  ';
OUT(127,CHR(153)];
PBLAKT:=0;
END;
PROCEDURE VYPISBLOKU;
VAR N:INTEGER;
BEGIN
PAGE;
FOR N:=POCPRIK+1 TO JMCELKEM DO BEGIN
 WRITE(CHR(20],CHR(1)];
 WRITE((N-POCPRIK]:2,CHR(20),CHR(0)];WR
ITE(JMENA[N]);END;
END;
PROCEDURE PROVADEN];
VAR VX,VY,SCALE:REAL;
    MFX,MFY,RYCHLOST,OTC:INTEGER;
    POCETKB:INTEGER;
    CKB:ARRAY[1..10] OF INTEGER;
    N:INTEGER;
BEGIN
REPEAT
INIT;
PAGE;
REPEAT
WRITELN('Loading file for drawing'];WRI
TELN;WRITELN;WRITELN;
PRLOAD;
WRITELN('All files loaded ? (Y/N) '];
REPEAT UNTIL INCH<>CHR(0);
UNTIL INCH='Y';
OUT(127,CHR(153));
WRITELN('Parameirs of drawing'];WRITELN
;
WRITE('Original point X.. ');READLN;REA
D(VX);
WRITE('              Y.. ');READLN;REA
D(VY);
WRITELN;WRITE('Scale ..... ');RE
ADLN;READ(SCALE);
WRITELN;WRITE('Orientation ..... ');RE
ADLN;READ(OTC);
WRITELN;WRITE('Speed [%] ..... ');RE
ADLN;
READ(RYCHLOST);WRITELN;WRITELN('Set or
ginal point  (0..exit]');
POKE(*EA78,CHR(223));POKE(*EA7C,CHR(215
)];POKE(*EA80,CHR(207];
POKE(*EA84,CHR(199));
IF OTC=3 THEN BEGIN POKE(*EA78,CHR(215)
);POKE(*EA7C,CHR(223));END;
IF OTC=2 THEN BEGIN POKE(*EA78,CHR(207)
);POKE(*EA7C,CHR(199)];
        POKE(*EA80,CHR(215]);POKE(*EA
84,CHR(223]];END;
USER (*EC54);
MF;
MFX:=PEEK(*EC66,INTEGER);
MFY:=PEEK((MFX+21,INTEGER];MFX:=PEEK(MF
X,INTEGER];
OG(MFX-SCALE*VX,MFY-SCALE*VY);
SC(SCALE);SPEED(ROUND(12300/RYCHLOST));
VYPISBLOKU;WRITELN;WRITELN;
REPEAT
WRITE('Nuber of blocks .. '];READLN;REA
D(POCETKB);
UNTIL (POCETKB>0) AND (POCETKB<11);
WRITELN;WRITELN;
FOR N:=1 TO POCETKB DO BEGIN
 AT(20,0];WRITE('
');
 AT(20,0];WRITE(N,'.... ');READLN;READ(
CKB[N]);
 END;
WRITELN;WRITELN('I am drawing');WRITELN
;
FOR N:=1 TO POCETKB DO BEGIN
 PBLOK.X:=0;PBLOK.Y:=0;PBLOK.CBL:=CHR(C
KB[N]);
 BLOKKRES(PBLOK);
 END;
WRITELN('Load a new file (Y/N)');
REPEAT UNTIL INCH<>CHR(0);
UNTIL INCH='N';
END;
BEGIN
 PROVADEN];
END.
```

BT 100 – ZX SPECTRUM

Obslužný program pro spolupráci tiskárny BT 100 s počítačem ZX Spectrum

Ing. Jiří Kohout, *Na Švihance 12, 120 00 Praha 2*

V příručce dodávané s tiskárnou BT 100 je uveden i program pro její obsluhu. Délka tohoto programu je přibližně 1kB, dalších 500 bajtů zabírá vyrovnávací paměť tiskárny. Umožňuje použití příkazů LLIST a LPRINT a kopii obrazovky. Program však neumí tisknout implicitně definované grafické znaky a znaky definované uživatelem v oblasti UDG a v příkazu LPRINT nemá příkaz TAB pro tabelaci výpisu účinek. Proto byl vytvořen program, který uvedené nedostatky odstraňuje a navíc je podstatně kratší.

Program pro spolupráci tiskárny BT 100 s mikropočítačem ZX Spectrum je dlouhý 706 bajtů, používá standardních příkazů LLIST a LPRINT (LPRINT i ve spojení s příkazem TAB), tiskne všechny grafické znaky, písmo lze zvolit s normálním sklonem nebo šikmé. Typ písma může být změněn uložením vlastního souboru znaků do RAM a příslušnou změnou systémové proměnné CHARS (zde je uložena adresa počátku tabulky znaků minus 256). Ukázka různých typů písma je na **obr. 1.** Program též umožňuje kopii obrazovky, příklad je na **obr. 2.** Jako vyrovnávací paměť pro tiskárnu je využita část obrazové paměti (horní třetina obrazovky), která při obsluze tiskárny většinou nebývá využita; navíc je na obrazovce vidět, co se právě tiskne.

Tiskárna je připojena k počítači přes rozhraní s obvodem 8255 (zapojení bylo v AR již několikrát publikováno, lze též použít výrobek Tesly Kolín, který je prodáván pod označením UR-4). Pro komunikaci počítač–tiskárna je použit kanál C, kde bity C0 až C3 jsou naprogramovány jako výstupní a bity C4 až C7 jako vstupní.

Nyní stručně k vlastnímu programu (viz **Výpis 1).** Řádky 140 až 250 (rutina INIT) naprogramují obvod 8255, do tabulky kanálových informací na adresu 23749 (#5CC5) uloží startovací adresu obslužného programu (při prvním vstupu adresu START1, kde se vymaže horní třetina obrazovky, při dalších vstupech pak adresu START) a nastaví počitadlo řádků.

Řádky 450 až 1060 rozliší podle kódu v registru A o jaký typ znaku se jedná a do registrového páru DE uloží adresu prvního bajtu znaku.

Kódy menší než 32 jsou ignorovány s výjimkou hodnoty 13 – konec řádku, 23 – TAB a 20 – INVERSE. Kódy 32 až 127 jsou standardní znaky ASCII, adresa prvního bajtu znaku se vytvoří na řádcích 1030 až 1060. Kódy 128 až 143 reprezentují implicitně definované grafické znaky, znak se tvoří pomocí rutiny v ROM začínající na adrese #0B38. Znak je pak uložen v systémové proměnné MEMBOT, kde začíná zásobníková paměť kalkulátoru. Tato rutina je na řádcích 730 až 750. Kódy 144 až 164 určují znaky definované uživatelem v oblasti RAM, tuto oblast adresuje systémová proměnná UDG. Adresa prvního bajtu příslušného znaku je určena na řádcích 680 až 710. 165 až 255 jsou kódy pro tokens, tvorbu jednotlivých znaků zajišťuje rutina ROM na adrese #09F4 (řádky 600 až 610).

Rutina ULOZ na řádcích 1070 až 1280 uloží jednotlivé bajty znaků do obrazové paměti, která slouží jako vyrovnávací paměť pro tiskárnu.

Na řádcích 1190 až 1280 se kontroluje, zda není překročen povolený počet znaků na řádek.

Na řádku 1290 se rozhoduje, zda se znaky budou tisknout normálně nebo šikmo. Při tisku šikmých znaků se před vstupem do vlastní tiskové rutiny provede posun jednotlivých bitů ve vyrovnávací paměti o 1 až 7 tiskových bodů vpravo – řádky 1520 až 1760.

Na řádcích 1320 až 1500 je volána tisková rutina TISK, po vytisknutí celého řádku je vynulována vyrovnávací paměť a je testován počet vytisknutých řádků. Při překročení předem nastaveného počtu se program zastaví a po stisknutí libovolné klávesy pokračuje v tisku – řádky 1400 až 1500. Vytisknutí celé stránky je signalizováno pípáním.

Rutina na řádcích 1780 až 1840 vypočítá adresu prvního bajtu znaku. Při vstupu do rutiny je v registrovém páru BC adresa začátku příslušné tabulky znaků (v ROM nebo v RAM) a v registru A je pořadové číslo znaku (první znak v tabulce má pořadové číslo nula).

Na řádcích 1870 až 1940 je vynulována vyrovnávací paměť.

Vlastní tisková rutina TISK začíná na řádku 1960. Nejprve se provede návrat vozíku a pak se testují vždy dva nad sebou ležící mikrořádky vyrovnávací paměti (jeden tiskový řádek se skládá z osmi mikrořádků). Pokud jsou oba nulové, posune se dvakrát papír v tiskárně a pokračuje se testem dalších dvou mikrořádků. Je-li alespoň jeden bajt v některém mikrořádku různý od nuly

Obr. 2. Ukázka kopie obrazovky (902–2)

(tzn. že je třeba tisknout), je v registrech D a E na řádku 2140 počet znaků zleva, které je třeba vytisknout. Vlastní test se provádí rutinou TEST na řádcích 3070 až 3200.

Řádky 2150 až 2630 vytisknou celý tiskový řádek (osm mikrořádků). Tiskne se střídavě lichý mikrořádek (při posunu vozíku zleva doprava) a sudý mikrořádek (při posunu vozíku zprava doleva).

Stiskne-li se tlačítko Q, tisk se zastaví po vytisknutí celého tiskového řádku a provede se skok do BASIC (řádky 2640 až 2690). Změnou hodnoty na řádku 2650 (v instrukci CP 81) lze změnit klávesu, kterou se zastavuje tisk.

Tisk jednoho bajtu provádí rutina na řádcích 2740 až 2930. Bajt je rotován do indikátoru CY buď vpravo nebo vlevo v závislosti na směru tisku (řádek 2750).

Rutina CR na řádcích 2950 až 3050 provede návrat vozíku, rutina PS_PAP (řádky 3220 až 3320) posune papír v tiskárně o jeden mikrořádek.

Rutiny C60 (řádky 3340 až 3400) a C70 (řádky 3420 až 3600) synchronizují tisk. O poloze vozíku tiskárny informují bity 7 a 6 na portu 95. Je-li bit 7 roven nule, vytiskne se jeden bod. Bit 6 slouží k určení začátku mikrořádku při tisku zprava i zleva.

Rutina na řádcích 3620 až 3660 jemně nastavuje synchronizaci tisku svislých čar.

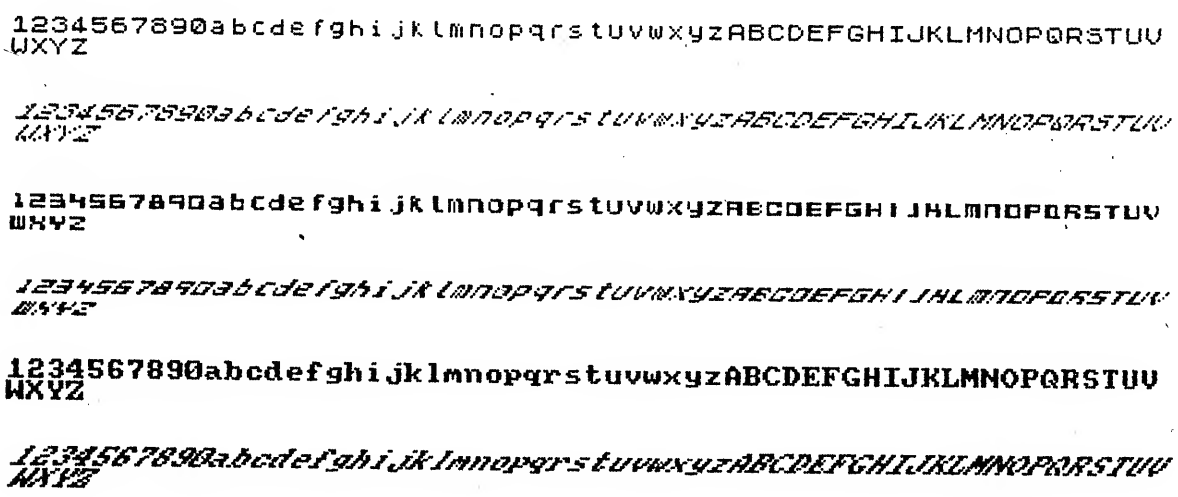
Změnou hodnoty v registru A. popřípadě přidáním dalších instrukcí NOP lze nastavit přesnou polohu vytisknutých bodů. Záleží na rychlosti posuvu vozíku, uvedené hodnoty vyhovovaly při rychlosti tisku 170 bodů/s. Rychlost vozíku lze nastavit trimrem na desce motorů (deska je umístěna podél přední strany tiskárny).

Obrazovku kopíruje program COPY na řádcích 280 až 400. Nejprve se provede inicializace tiskárny a nastaví se šířka tisku na 32 znaků/řádek. Potom se postupně vytiskne všech 24 tiskových řádků obrazovky (vždy po vytisknutí jednoho tiskového řádku rutina POSUV na řádcích 3680 až 4090 posune do bafru další tiskový řádek), konec tisku je signalizován pípáním a po stisku libovolné klávesy se provede návrat do BASICu. Je-li program nahrán od adresy A (v uvedeném případě A=64000), inicializace se provede příkazem **RANDOMIZE USR A,** pak již mohou následovat BASIC příkazy LLIST nebo LPRINT. Formát tisku lze nastavit uložením příslušných hodnot na následující adresy:

počet řádků na stránku	**POKE A+22, počet řádků**
počet znaků na řádek	**POKE A+216, počet znaků**
písmo normal	**POKE A+225,0**
písmo šikmé	**POKE A+225,46**
synchronizace tisku	**POKE A+626,hodnota**

Obrazovka se zkopíruje příkazem **RANDOMIZE USR A+2.**

Program spolupracuje i s assemblerem GENS3, obslužný program pro tiskárnu pak lze zkrátit asi na 540 bajtů vypuštěním částí pro kopírování obrazovky, tisk grafických znaků a tisk šikmého písma. Konkrétně se



Obr. 1. Ukázka různých typů písma (902–1)

vypustí řádky 40, 50, 110, 120, 230–250, 280–400, 460–530, 560–590, 640–1010, 1520–1760, 3680–4090, 4130 a 4140. Po tomto zkrácení programu se však změní výše uvedené adresy pro nastavení formátu tisku; úpravu si provede již jistě každý sám.

Výpis 1. *Obslužný program* (902–V1)

```
                10 *C-
                20 ;BT 100 pro LLIST,
                30
5C7B    40 UDG      EQU  23675
5C92    50 MEMBOT   EQU  23698
5C36    60 CHARS    EQU  23606
5C08    70 LAST_K   EQU  23560
4000    80 BAFR     EQU  #4000
                90
FA00   100          ORG  64000
FA00   110          JR   INIT
FA02   120          JR   COPY
                130
FA04   140 INIT     CALL UL_BFR
FA07   150          LD   A,154
FA09   160          OUT  (#7F),A
FA0B   170          LD   A,255
FA0D   180          OUT  (95),A
FA0F   190          LD   HL,START1
FA12   200          LD   (#5CC5),HL
FA15   210 INIRAD   LD   A,75
                    ;pocet radku
FA17   220          LD   (P_RAD),A
FA1A   230 NUL_AD   XOR  A
FA1B   240          LD   (AD_TAB),A
FA1E   250          LD   (AD_VID),A
FA21   260          RET
                270
FA22   280 COPY     CALL INIT
FA25   290          LD   A,32
FA27   300          LD   (R_COPY),A
FA2A   310          LD   B,24
FA2C   320 TCOPY    PUSH BC
FA2D   330          CALL TISK
FA30   340          CALL POSUV
FA33   350          POP  BC
FA34   360          DJNZ TCOPY
FA36   370          LD   A,59
FA38   380          LD   (R_COPY),A
FA3B   390          CALL KEY
FA3E   400          RET
                410
FA3F   420 START1   CALL NULBFR
FA42   430          LD   HL,START
FA45   440          LD   (23749),HL
FA48   450 START    EI
FA49   460          LD   E,A
FA4A   470          LD   A,(AD_TAB)
FA4D   480          CP   1
FA4F   490          JR   Z,TAB1
FA51   500          LD   A,(AD_VID)
FA54   510          CP   2
FA56   520          JR   Z,VID1
FA58   530          LD   A,E
FA59   540          CP   13
FA5B   550          JP   Z,TSK
FA5E   560          CP   #17
FA60   570          JR   Z,TAB
FA62   580          CP   20
FA64   590          JR   Z,VID
FA66   600          CP   165
FA68   610          JP   NC,#09F4
FA6B   620          CP   32
FA6D   630          RET  C
FA6E   640          CP   128
FA70   650          JR   C,ZNAK
FA72   660          CP   144
FA74   670          JR   C,IMP_GR
FA76   680          LD   BC,(UDG)
FA7A   690          SUB  144
FA7C   700          CALL ADR_ZN
FA7F   710          JR   ULOZ
                720
FA81   730 IMP_GR   LD   B,A
FA82   740          CALL #0838
FA85   750          LD   DE,MEMBOT
FA88   760          JR   ULOZ
                770
FA8A   780 TAB      LD   A,1
FA8C   790          LD   (AD_TAB),A
FA8F   800          RET
                810
FA90   820 TAB1     LD   D,0
FA92   830          LD   A,E
FA93   840          CP   59
FA95   850          JR   NC,TSK
FA97   860          LD   HL,BAFR
FA9A   870          ADD  HL,DE
FA9B   880          LD   (AD_BFR),HL
FA9E   890          JP   NUL_AD
                900
FAA1   910 VID      LD   A,2
FAA3   920          LD   (AD_VID),A
FAA6   930          RET
                940
FAA7   950 VID1     LD   HL,AD_INV
FAAA   960          LD   A,E
FAAB   970          CP   1
FAAD   980          JR   NZ,TRUE
FAAF   990          LD   A,47
FAB1  1000 TRUE     LD   (HL),A
FAB2  1010          JP   NUL_AD
               1020
FAB5  1030 ZNAK     LD   BC,(CHARS)
FAB9  1040          INC  B
FABA  1050          SUB  32
FABC  1060          CALL ADR_ZN
FABF  1070 ULOZ     LD   B,8
FAC1  1080          LD   HL,(AD_BFR)
FAC4  1090          PUSH HL
FAC5  1100 ULOZ1    LD   A,(DE)
FAC6  1110 AD_INV   DEFB 0
                    ;NOP nebo CPL
FAC7  1120          LD   (HL),A
FAC8  1130          INC  DE
FAC9  1140          INC  H
FACA  1150          DJNZ ULOZ1
FACC  1160          POP  HL
FACD  1170          INC  HL
FACE  1180          LD   (AD_BFR),HL
FAD1  1190          PUSH HL
FAD2  1200          LD   HL,BAFR
FAD5  1210          LD   D,0
FAD7  1220          LD   E,58
                    ;pocet znaku
FAD9  1230          DEC  E
FADA  1240          ADD  HL,DE
FADB  1250          POP  DE
FADC  1260          AND  A
FADD  1270          SBC  HL,DE
FADF  1280          RET  NC
FAE0  1290 TSK      JR   T_SK1
      1300 ;nebo JR ROTACE
      1310
FAE2  1320 T_SK1    CALL TISK
FAE5  1330          CALL NULBFR
FAE8  1340          LD   A,(P_RAD)
FAEB  1350          DEC  A
FAEC  1360          LD   (P_RAD),A
FAEF  1370          CP   0
FAF1  1380          RET  NZ
FAF2  1390          CALL INIRAD
FAF5  1400 KEY      LD   DE,157
FAF8  1410          LD   HL,10085
FAFB  1420          CALL #03B5
FAFE  1430          LD   DE,411
FB01  1440          LD   HL,715
FB04  1450          CALL #03B5
FB07  1460          CALL #028E
FB0A  1470          LD   A,E
FB0B  1480          CP   255
FB0D  1490          JR   Z,KEY
FB0F  1500          RET
               1510
FB10  1520 ROTACE   LD   HL,#463A
FB13  1530 ROT1     PUSH HL
FB14  1540 ROT2     PUSH HL
FB15  1550          LD   B,59
FB17  1560 ROT3     LD   A,(HL)
FB18  1570          SRL  A
FB1A  1580          LD   (HL),A
FB1B  1590          JR   NC,ROT4
FB1D  1600          INC  HL
FB1E  1610          LD   A,(HL)
FB1F  1620          SET  7,A
FB21  1630          LD   (HL),A
FB22  1640          DEC  HL
FB23  1650 ROT4     DEC  HL
FB24  1660          DJNZ ROT3
FB26  1670          POP  HL
FB27  1680          DEC  H
FB28  1690          LD   A,#3F
FB2A  1700          CP   H
FB2B  1710          JR   NZ,ROT2
FB2D  1720          POP  HL
FB2E  1730          DEC  H
FB2F  1740          CP   H
FB30  1750          JR   NZ,ROT1
FB32  1760          JR   T_SK1
               1770
FB34  1780 ADR_ZN   LD   H,0
FB36  1790          LD   L,A
FB37  1800          ADD  HL,HL
FB38  1810          ADD  HL,HL
FB39  1820          ADD  HL,HL
FB3A  1830          ADD  HL,BC
FB3B  1840          EX   DE,HL
FB3C  1850          RET
               1860
FB3D  1870 NULBFR   LD   HL,BAFR
FB40  1880          LD   DE,BAFR+1
FB43  1890          LD   (HL),0
FB45  1900          LD   BC,2047
FB48  1910          LDIR
FB4A  1920 UL_BFR   LD   HL,BAFR
FB4D  1930          LD   (AD_BFR),HL
FB50  1940          RET
               1950
FB51  1960 TISK     CALL CR
FB54  1970          LD   HL,BAFR
FB57  1980          LD   B,4
FB59  1990 SMYC     PUSH BC
FB5A  2000          CALL TEST
FB5D  2010          LD   HL,(AD_BFR)
FB60  2020          INC  H
FB61  2030          LD   C,E
FB62  2040          CALL TEST
FB65  2050          LD   HL,(AD_BFR)
FB68  2060          DEC  H
FB69  2070          LD   A,C
FB6A  2080          OR   E
FB6B  2090          JP   Z,OBA_0
FB6E  2100          LD   A,C
FB6F  2110          CP   E
FB70  2120          JR   C,MENSI
FB72  2130          LD   E,C
FB73  2140 MENSI    LD   D,E
FB74  2150          LD   A,7
FB76  2160          LD   (AD_RL),A
FB79  2170          LD   A,253
FB7B  2180          OUT  (95),A
FB7D  2190          CALL C70
FB80  2200          LD   B,10
FB82  2210 SM_POC   CALL C70
FB85  2220          DJNZ SM_POC
FB87  2230 VPRAVO   LD   A,(HL)
FB88  2240          CALL T_TISK
FB8B  2250          INC  HL
FB8C  2260          DEC  E
FB8D  2270          JR   NZ,VPRAVO
FB8F  2280          LD   B,255
FB91  2290 R1       CALL C70
FB94  2300          INC  B
FB95  2310          IN   A,(95)
FB97  2320          AND  64
FB99  2330          JR   NZ,P1
FB9B  2340          CALL C70
FB9E  2350          CALL C70
FBA1  2360          LD   A,255
FBA3  2370          OUT  (95),A
FBA5  2380 LICHY    CALL PS_PAP
FBA8  2390          LD   E,D
FBA9  2400          LD   D,0
FBAB  2410          DEC  E
FBAC  2420          LD   HL,(AD_BFR)
FBAF  2430          ADD  HL,DE
FBB0  2440          INC  E
FBB1  2450          LD   A,15
FBB3  2460          LD   (AD_RL),A
FBB6  2470          LD   A,247
FBB8  2480          OUT  (95),A
FBBA  2490          CALL C60
FBBD  2500 L1       CALL C70
FBC0  2510          DJNZ L1
FBC2  2520 VLEVO    LD   A,(HL)
FBC3  2530          CALL T_TISK
FBC6  2540          DEC  HL
FBC7  2550          DEC  E
FBC8  2560          JR   NZ,VLEVO
FBCA  2570          CALL CR1
FBCD  2580 SUDY     CALL PS_PAP
FBD0  2590          LD   HL,(AD_BFR)
FBD3  2600          INC  H
FBD4  2610          POP  BC
FBD5  2620          DEC  B
FBD6  2630          JP   NZ,SMYC
FBD9  2640          LD   A,(LAST_K)
FBDC  2650          CP   81
FBDE  2660          RET  NZ
FBDF  2670          CALL INIT
FBE2  2680          RST  8
FBE3  2690          DEFB 255
      2700
FBE4  2710 OBA_0    CALL PS_PAP
FBE7  2720          JR   SUDY
      2730
FBE9  2740 T_TISK   LD   B,8
FBEB  2750 AD_RL    DEFB 0
      2760 ;RLCA nebo RRCA
      2770
FBEC  2780          LD   C,A
FBED  2790          LD   A,0
FBEF  2800          ADC  A,A
FBF0  2810          PUSH AF
FBF1  2820          CALL C70
FBF4  2830          POP  AF
FBF5  2840          OR   A
FBF6  2850          JR   Z,T_BIT1
FBF8  2860          IN   A,(95)
FBFA  2870          AND  254
FBFC  2880          OUT  (95),A
FBFE  2890          OR   1
FC00  2900          OUT  (95),A
FC02  2910 T_BIT1   LD   A,C
FC03  2920          DJNZ AD_RL
FC05  2930          RET
      2940
FC06  2950 CR       IN   A,(95)
FC08  2960          AND  32
FC0A  2970          RET  NZ
FC0B  2980          LD   A,247
FC0D  2990          OUT  (95),A
FC0F  3000 CR1      IN   A,(95)
FC11  3010          AND  32
FC13  3020          JR   Z,CR1
FC15  3030          LD   A,255
FC17  3040          OUT  (95),A
FC19  3050          RET
      3060
FC1A  3070 TEST     LD   (AD_BFR),HL
FC1D  3080          DEFB 17
FC1E  3090 R_COPY   DEFB 59
FC1F  3100          DEFB 0    ;LD DE,59
      3110 ;nebo LD DE,32 pro COPY
      3120
FC20  3130          ADD  HL,DE
FC21  3140 TST1     DEC  HL
FC22  3150          LD   A,(HL)
FC23  3160          OR   A
FC24  3170          RET  NZ
FC25  3180          DEC  E
FC26  3190          RET  Z
FC27  3200          JR   TST1
      3210
FC29  3220 PS_PAP   LD   A,251
FC2B  3230          OUT  (95),A
FC2D  3240 PS1      IN   A,(95)
FC2F  3250          AND  16
FC31  3260          JR   Z,PS1
FC33  3270 PS2      IN   A,(95)
FC35  3280          AND  16
FC37  3290          JR   NZ,PS2
FC39  3300          LD   A,255
FC3B  3310          OUT  (95),A
FC3D  3320          RET
      3330
FC3E  3340 C60      IN   A,(95)
FC40  3350          AND  64
FC42  3360          JR   NZ,C60
FC44  3370          IN   A,(95)
FC46  3380          AND  64
FC48  3390          JR   NZ,C60
FC4A  3400          RET
      3410
FC4B  3420 C70      IN   A,(95)
FC4D  3430          AND  128
FC4F  3440          JR   Z,C70
FC51  3450          IN   A,(95)
FC53  3460          AND  128
FC55  3470          JR   Z,C70
FC57  3480 C71      IN   A,(95)
FC59  3490          AND  128
FC5B  3500          JR   NZ,C71
FC5D  3510          IN   A,(95)
FC5F  3520          AND  128
FC61  3530          JR   NZ,C71
FC63  3540          IN   A,(95)
FC65  3550          AND  128
FC67  3560          JR   NZ,C71
FC69  3570          IN   A,(95)
FC6B  3580          AND  128
FC6D  3590          JR   Z,CEKEJ
FC6F  3600          JR   C71
      3610
FC71  3620 CEKEJ    LD   A,200
      ;synchronizace tisku
FC73  3630 CEK      NOP
FC74  3640          DEC  A
FC75  3650          RET  Z
FC76  3660          JR   CEK
      3670
FC78  3680 POSUV    LD   B,8
FC7A  3690 PREN     PUSH BC
FC7B  3700          LD   DE,#5B00
FC7E  3710          PUSH DE
FC7F  3720          LD   HL,#4000
FC82  3730          LD   B,24
FC84  3740 P_OBR    PUSH BC
FC85  3750          LD   B,8
FC87  3760 PREN1    PUSH BC
FC88  3770          PUSH HL
FC89  3780          PUSH HL
FC8A  3790          LD   BC,32
FC8D  3800          LDIR
FC8F  3810          POP  HL
FC90  3820          INC  H
FC91  3830          POP  DE
FC92  3840          POP  BC
FC93  3850          DJNZ PREN1
FC95  3860          POP  BC
FC96  3870          DEC  B
FC97  3880          PUSH DE
FC98  3890          CALL #0E9B
FC9B  3900          POP  DE
FC9C  3910          INC  B
FC9D  3920          DJNZ P_OBR
FC9F  3930          POP  HL
FCA0  3940          LD   BC,32
FCA3  3950          LDIR
FCA5  3960          POP  BC
FCA6  3970          DJNZ PREN
FCA8  3980          EX   DE,HL
FCA9  3990          LD   DE,#5800
FCAC  4000          LD   BC,32
FCAF  4010          PUSH BC
FCB0  4020          PUSH HL
FCB1  4030          LDIR
FCB3  4040          POP  DE
FCB4  4050          LD   BC,736
FCB7  4060          LDIR
FCB9  4070          POP  BC
FCBA  4080          LDIR
FCBC  4090          RET
      4100
FCBD  4110 AD_BFR   DEFB 0,0
FCBF  4120 P_RAD    DEFB 0
FCC0  4130 AD_TAB   DEFB 0
FCC1  4140 AD_VID   DEFB 0
      4150
```